

UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE QUITO-CAMPUS SUR

CARRERA DE INGENIERÍA DE SISTEMAS

MENCIÓN ROBOTICA E INTELIGENCIA ARTIFICIAL

**IMPLEMENTACIÓN DE UN ALGORITMO DE BÚSQUEDA
INFORMADA EN EL ROBOT MÓVIL ROBOTINO DE FESTO PARA
LA OBTENCIÓN DE LA TRAYECTORIA MÁS ÓPTIMA EN TIEMPO
REAL DENTRO DE UN ENTORNO CONTROLADO**

TESIS PREVIA A LA OBTENCIÓN DEL TÍTULO DE INGENIERO DE SISTEMAS

GUALOTUÑA VILLAVICENCIO ROBERTO JAIRO

LLININ REA EDISSON ORLANDO

DIRECTOR ING. RODRIGO TUFIÑO

Quito, Julio 2011

DECLARACIÓN

Nosotros, Gualotuña Villavicencio Roberto Jairo y Llinin Rea Edison Orlando, declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que hemos consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedemos nuestros derechos de propiedad intelectual correspondientes a este trabajo, a la Universidad Politécnica Salesiana, según lo establecido por la Ley de Propiedad Intelectual, por su reglamento y por la normatividad institucional vigente.

Gualotuña Villavicencio Roberto J.

Llinin Rea Edison O.

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Gualotuña Villavicencio Roberto Jairo y Llinin Rea Edison Orlando bajo mi dirección.

Ing. Rodrigo Tufiño
Director de tesis

AGRADECIMIENTOS

A todos nuestros maestros de la Carrera de Ingeniería de Sistemas, en especial al Ing. Rodrigo Tufiño, director del presente trabajo, por su presencia incondicional, sus apreciados y relevantes aportes, críticas, comentarios y sugerencias durante el desarrollo de esta investigación.

A nuestra lectora de tesis; Ing. Doris Bautista, por sus valiosas correcciones que ayudaron a mejorar este trabajo, y por su amistad.

Al Ing. Germán Arévalo, director de la Carrera de Ingeniería Electrónica, por permitirnos utilizar los laboratorios en todas las ocasiones que fueron necesarios.

Finalmente, agradecemos a los encargados de los laboratorios de electrónica, por su infinita paciencia día a día hasta culminar la aplicación mencionada en el presente proyecto de tesis.

AGRADECIMIENTOS

Primeramente quiero dar gracias a Dios, por haberme dado la fuerza y el valor para culminar mis estudios.

Agradezco la confianza y el apoyo brindado por mis padres Néstor Gualotuña y Sabina Villavicencio, mi hermano Alex Gualotuña por creer en mí, y no dejarme caer nunca apoyándome continuamente a que realice este proyecto de tesis con mucha más ilusión.

A mis abuelos y tíos por sus sabios consejos que me enseñaron a nunca rendirme y seguir siempre hacia delante.

Por ultimo también quería mostrar un especial agradecimiento a todos mis maestros que durante la carrera universitaria, compartieron su conocimiento, ayudándome a crecer como persona y como profesional.

A todos ellos: Gracias

Roberto Gualotuña

AGRADECIMIENTOS

Primero y antes de nada, dar gracias a DIOS, por estar conmigo en cada paso que doy, por fortalecer mi corazón e iluminar mi mente.

Agradezco a mis padres Gilberto Llinin y María Rea de todo corazón por su amor, cariño y comprensión, a pesar de mi ausencia y malos momentos.

También quiero agradecer a mi hermano Braulio Llinin por el apoyo y ánimo que me brindó, quien ha sido un soporte muy fuerte en momentos de angustia y desesperación.

A mi compañero de tesis, por su especial apoyo, su comprensión en muchas ocasiones y por todos los momentos que hemos pasado juntos, por todas las conversaciones y buenos momentos de amistad y compañerismo.

Gracias también a mis queridos amigos y familia importantes en mi vida, a quienes siempre tengo presente, muchas gracias por estar allí en el momento y tiempo exactos.

Gracias a todos.

Edisson Llinin

CONTENIDO

CAPÍTULO 1.....	1
1.1 INTRODUCCIÓN.....	1
1.2 OBJETIVO GENERAL.....	2
1.2.1 OBJETIVOS ESPECÍFICOS	2
1.3 ALCANCE DEL PROYECTO.....	2
1.4 PLANTEAMIENTO DEL PROBLEMA	3
CAPÍTULO 2.....	4
2.1 ROBOTS MÓVILES.....	4
2.1.1 SISTEMAS DE LOCOMOCIÓN PARA ROBOTS MÓVILES	5
2.1.2 CONFIGURACIÓN CINEMÁTICA DE LOS ROBOTS MÓVILES CON RUEDAS	8
2.1.2.1 CONFIGURACIÓN ACKERMAN	8
2.1.2.2 CONFIGURACIÓN TRICICLO.....	9
2.1.2.3 CONFIGURACIÓN DIFERENCIAL.....	10
2.1.2.4 CONFIGURACIÓN SÍNCRONA.....	11
2.1.2.5 CONFIGURACIÓN OMNIDIRECCIONAL	11
2.2 PERCEPCIÓN.....	13
2.2.1 MICROINTERRUPTORES DE CHOQUE O BUMPERS	14
2.2.2 SENSORES DE PRESENCIA Y PROXIMIDAD.....	14
2.2.3 SENSORES INDUCTIVOS.....	15
2.2.4 SENSORES ÓPTICOS	16
2.2.5 ENCODERS	17
2.2.5.1 ENCODERS INCREMENTALES.....	17
2.3 ACTUADORES	18
2.4 ARQUITECTURA PARA EL CONTROL DE ROBOTS MÓVILES	20
2.4.1 CONTROL INTELIGENTE	20
2.4.2 REQUERIMIENTOS GENERALES DE LA ARQUITECTURA.....	20
2.4.3 TIPOS DE ARQUITECTURAS DE CONTROL	22
2.4.4 TIPOS DE AMBIENTES	24
2.4.5 NAVEGACIÓN EN ROBOTS MÓVILES.....	25
2.4.6 ESQUEMA DE NAVEGACIÓN EN UN ROBOT MÓVIL	27
2.4.7 APLICACIONES	29
2.4.8 APLICACIONES ESPECIALES	32

2.5	INTELIGENCIA ARTIFICIAL	37
2.5.1	RESOLUCIÓN DE PROBLEMAS MEDIANTE BÚSQUEDAS.....	38
2.5.2	ESPACIO DE ESTADOS	39
2.5.3	ESTRATEGIAS DE BÚSQUEDA	41
2.5.3.1	ESTRATEGIAS DE BÚSQUEDA NO INFORMADA (CIEGAS)	42
2.5.3.2	ESTRATEGIAS DE BÚSQUEDA INFORMADA (HEURÍSTICAS) ..	44
3.1	ANÁLISIS.....	61
3.1.1	ROBOTINO	61
3.1.1.1	CHASIS Y BATERÍAS	63
3.1.1.2	UNIDAD DE ACCIONAMIENTO	64
3.1.1.3	SENSORES	65
3.1.1.4	PUENTE DE MANDO	70
3.1.1.5	UNIDAD DE CONTROL	71
3.1.1.6	TECLADO DE MEMBRANA Y DISPLAY	71
3.1.1.7	TARJETA COMPACT FLASH.....	72
3.1.1.8	INTERFACE E/S	73
3.1.1.9	MÓDULO TARJETA CIRCUITO DE ENTRADA	74
3.1.1.10	PUNTO DE ACCESO LAN INALÁMBRICO	74
3.1.1.11	LABTEC WEBCAM PRO	75
3.1.2	API (INTERFAZ DE PROGRAMACIÓN DE APLICACIONES).....	76
3.1.3	PRUEBAS DE LOS SENSORES	82
3.2	DISEÑO	84
3.2.1	DIAGRAMAS DE CASOS DE USO	84
3.2.2	DISEÑO DEL ENTORNO DE TRABAJO	89
3.2.3	ARQUITECTURA DEL SISTEMA DE CONTROL	91
3.3	DISEÑO DEL ALGORITMO	92
3.4	INTERACCIÓN CON EL ENTORNO Y CARACTERÍSTICAS EN TIEMPO REAL.....	93
3.5	DISEÑO DE LA INTERFAZ	96
CAPÍTULO 4	99
4.1	DIAGRAMA DE CLASES	99
4.2	PROGRAMACIÓN DE SOFTWARE	101
4.3	HERRAMIENTAS DE SOFTWARE UTILIZADAS	114
4.4	IMPLEMENTACIÓN DEL ALGORITMO	116
4.5	PRUEBAS REALIZADAS	120
4.5.1	PRUEBAS DE SOFTWARE	120
4.5.2	PRUEBAS SOBRE EL ROBOTINO.....	121
4.6	CONDICIONES DEL ENTORNO DE TRABAJO.....	124

4.7 RESULTADOS OBTENIDOS.....	126
CONCLUSIONES Y RECOMENDACIONES	128
REFERENCIAS BIBLIOGRÁFICAS	131
ANEXO 1.....	A-1
ANEXO 2.....	A-5
ANEXO 3.....	A-34
ANEXO 4.....	A-61

ÍNDICE DE FIGURAS

Figura 2.1: Irobot 710 Warrier (iROBOT).....	6
Figura 2.2: Asimo – robot humanoide (HONDA)	7
Figura 2.3: Aibo – es un robot mascota (SONY).....	7
Figura 2.4: Robot hexápodo Silo6 (Department of Automatic Control, Industrial Automation Institute and Institute Spanish National Research Council	7
Figura 2.5: The ATHLETE Rover (NASA) Fuente: [Robot hexápodo – Athlete Rover] ...	7
Figura 2.6: Robot Nomad desarrollado por investigadores del departamento de robótica de la Universidad Carnegie Mellon, con financiación de la NASA.	8
Figura 2.7: Posición de las ruedas en configuración Ackerman	9
Figura 2.8: Configuración triciclo.....	10
Figura 2.9: Disposición de las ruedas en configuración diferencial con una rueda loca al frente	10
Figura 2.10: Configuración Síncrona con tres ruedas	11
Figura 2.11: a) Robot omnidireccional Rovio (WowWee), b) Uranu mobile robot (Carnegie- Mellon University).	12
Figura 2.12: Tipos de ruedas para robots móviles	12
Figura 2.13: Microinterruptor de choque o bumper	14
Figura 2.14: a) Sensor Inductivo, b) Componentes de un sensor Inductivo	16
Figura 2.15: a) Detector óptico de reflexión, b) Detector óptico de barrera	17
Figura 2.16: Encoder Incremental	18
Figura 2.17: Arquitectura Reactiva	23
Figura 2.18: Arquitectura Deliberativa	23
Figura 2.19: Arquitectura Híbrida.....	24
Figura 2.20: Esquema básico de la arquitectura necesaria en un robot móvil para realizar una misión.	26
Figura 2.21: Navegador implantado en el robot móvil Blanche de AT&T.....	28
Figura 2.22: Agv Eagle E210 Flexcart (CORECON)	30
Figura 2.23: a) Robot de limpieza del hogar Roomba 531 (iRobot), b) Robot de limpieza de piscinas Solar-Breeze (Eco Pool Technologies).....	30
Figura 2.24: Robot Trenz-Clenear (Trenz-Clenear).....	31
Figura 2.25: Robot mSecurit (MoviRobotics)	31
Figura 2.26: Yurina – Robot asistencial japonés (Toyota)	32

Figura 2.27: Robot militar R-Gator con kernelGNU/Linux (iRobot)	33
Figura 2.28: AIBO modelo ERS-7M3 (Sony)	34
Figura 2.29: a) Robots con ruedas en combate de sumo, b) Robots humanoides en combate de sumo	35
Figura 2.30: Robot en una prueba de laberinto	35
Figura 2.31: Robot en prueba de velocidad.....	36
Figura 2.32: Robot rastreador.....	36
Figura 2.33: a) Robots de plataforma estándar b) Robots humanoides	37
Figura 2.34: Un ejemplo típico del 8-puzzle.....	39
Figura 2.35: Representación de un árbol y sus elementos	40
Figura 2.36: Se muestra el proceso de búsqueda para llegar desde S a I utilizando el algoritmo Primero en Anchura.	43
Figura 2.37: La gráfica muestra el descenso realizado por el algoritmo de búsqueda Primero en profundidad de una manera más clara	44
Figura 2.38: Representación del coste total de la búsqueda frente al grado de conocimiento utilizado	45
Figura 2.39: a) Grafo, b) Distancia en línea recta al nodo 11	47
Figura 2.40: Etapas de la búsqueda primero el mejor para ir del nodo 1 al nodo 11	49
Figura 2.41: Condición de Consistencia	51
Figura 2.42: Redireccionamiento realizado en un nodo que se encuentra en ABIERTOS	55
Figura 2.43: Etapas de la búsqueda A* desde el nodo 1 al nodo 11	58
Figura 2.44: Etapas de la búsqueda A* desde el nodo 1 al nodo 11	60
Figura 3.1: Robotino de FESTO	61
Figura 3.2: Entorno 3D del Robotino (Robotino Sim).....	63
Figura 3.3: Chasis y componentes del Robotino.....	64
Figura 3.4: a) Ubicación de las unidades de accionamiento, b) Unidad de accionamiento y sus partes	65
Figura 3.5: Distribución de los sensores de distancia	66
Figura 3.6: Encoder incremental RE30	67
Figura 3.7: Sensor inductivo	67
Figura 3.8: Conexión del sensor inductivo a la interface E/S	68
Figura 3.9: Sensor óptico	69
Figura 3.10: Conexión eléctrica del sensor óptico	69

Figura 3.11: Curva de aproximación.....	70
Figura 3.12: Puente de mando del Robotino	70
Figura 3.13: Interfaz de E/S	71
Figura 3.14: Teclado de membrana y display	72
Figura 3.15: Tarjeta Compact Flash.....	73
Figura 3.16: Asignación de bornes del Interfaz de E/S.....	73
Figura 3.17: Tarjeta circuito de entrada	74
Figura 3.18: Comunicación a través de Wlan	75
Figura 3.19: Webcam del Robotino	76
Figura 3.20: Librería rec,robotino.com y sus respectivas clases.....	77
Figura 3.21: Tensión de salida del sensor inductivo ante diversos materiales.....	83
Figura 3.22: a) Imagen fuera de foco b) Imagen enfocada	83
Figura 3.23: Creación del entorno virtual	85
Figura 3.24: Control manual	86
Figura 3.25: Configuración de parámetros del Robotino	87
Figura 3.26: Configuración inicial del algoritmo.....	88
Figura 3.27: Navegación del Robotino	89
Figura 3.28: Diseño del entorno de trabajo	90
Figura 3.29: Cálculo de la distancia entre dos nodos.....	90
Figura 3.30: Arquitectura de Control Deliberativa para Robotino de FESTO.....	91
Figura 3.31: Proceso A*.....	93
Figura 3.32: Identificación del nodo objetivo	94
Figura 3.33: Seguidor inductivo.....	95
Figura 3.34: Límite interno y externo del camino.....	96
Figura 3.35: Ventana principal.....	97
Figura 3.36: Ventana de creación del entorno de trabajo	97
Figura 3.37: Ventana para crear el entorno	98
Figura 3.38: Ventana para la configuración de parámetros del Robotino.....	98
Figura 4.1: Diagrama de clases	100
Figura 4.2: Esquema del funcionamiento final del Sistema de Búsqueda	102
Figura 4.3: Cálculo del coste.....	103
Figura 4.4: Imagen del entorno virtual tipo bitmap	104
Figura 4.5: Diagrama de secuencia del Módulo Gráfico	105

Figura 4.6: Código de barras capturado por la webcam del Robotino	106
Figura 4.7: String con la identificación del código de barras.....	106
Figura 4.8: Diagrama de secuencia del Módulo BarcodeLib	107
Figura 4.9: Tablas utilizadas para la gestión de la información del entorno virtual	108
Figura 4.10: Archivo XML generado desde un dataSet por el Sistema de Búsqueda	108
Figura 4.11: Diagrama de secuencia del Módulo de Datos.....	110
Figura 4.12: Distancia heurística calculada desde el nodo objetivo hacia todos los nodos	111
Figura 4.13: Diagrama de secuencia del Módulo Inteligencia Artificial	112
Figura 4.14: Robotino en el proceso de seguimiento del camino	113
Figura 4.15: Robotino en el proceso de detección de caminos	113
Figura 4.16: Diagrama de secuencia del Módulo Robotino.....	114
Figura 4.17: Generación del código de barras Code128	115
Figura 4.18: Ubicación del Robotino utilizando el panel de control	117
Figura 4.19: Camino encontrado por el simulador para ir del nodo 6 al nodo 7.....	118
Figura 4.20: Camino seguido por el Robotino para ir del nodo 6 al nodo 7	119
Figura 4.21: Camino encontrado por el simulador para ir del nodo 16 al nodo 3.....	119
Figura 4.22: Camino seguido por el Robotino para ir del nodo 16 al nodo 3.....	120
Figura 4.23: Robotino en un camino menor a 65 cm	125

ÍNDICE DE TABLAS

Tabla 3.1: Clase AnalogInput.....	77
Tabla 3.2: Clase Bumper	78
Tabla 3.3: Clase Camara	78
Tabla 3.4: Clase Com	79
Tabla 3.5: Clase DigitalInput	79
Tabla 3.6: Clase DigitalOutput.....	80
Tabla 3.7: Clase DistanceSensor	80
Tabla 3.8: Clase EncoderInput	80
Tabla 3.9: Clase Motor	81
Tabla 3.10: Clase OmniDrive.....	82
Tabla 3.11: Comportamiento del sensor óptico frente a materiales de diferente color	82
Tabla 3.12: Tensión de salida del sensor inductivo ante diversos materiales	83
Tabla 4.1: Descripción de clases	99
Tabla 4.2: Pruebas de Software	121
Tabla 4.3: Pruebas sobre el Robotino	122
Tabla 4.4: Velocidades del Robotino	124

RESUMEN

Con los avances que se ha realizado en la robótica móvil se tienen en la actualidad robots móviles que realizan tareas de manera autónoma, y junto con el empleo de la Inteligencia Artificial se logra incrementar su grado de autonomía en tareas mucho más complejas como son: búsquedas, exploración, supervisión, etc.

El presente proyecto de tesis hace uso del robot móvil Robotino de FESTO para realizar tareas de búsqueda en tiempo real utilizando el algoritmo de búsqueda informada A*, con el fin de encontrar el camino de menor coste dentro de un entorno controlado. El sistema de control está basado en una arquitectura deliberativa, es decir, constituida de un nivel superior y un nivel inferior, las mismas que se comunican a través de una WLAN.

El nivel superior de la arquitectura de control se encuentra en un computador donde se ejecuta el algoritmo de búsqueda informada A* y el software para el control del Robotino desarrollados bajo la plataforma C# .NET, mientras que los caminos generados por el algoritmo de búsqueda son ejecutados por el nivel inferior de la arquitectura de control (hardware Robotino) a través de órdenes enviadas del nivel superior utilizando todas las librerías necesarias para el análisis de la información proveniente del entorno de trabajo, llevando de esta manera al robot móvil Robotino de FESTO por los caminos más eficientes al momento de desplazarse desde un estado inicial hacia un objetivo.

El sistema de búsqueda ha sido desarrollado con dos funcionalidades, utilizando el robot móvil Robotino de FESTO o independiente de él, con el fin de observar la eficiencia del algoritmo de búsqueda informada con otro tipo de entornos mucho más complejos.

PRESENTACIÓN

El presente proyecto de titulación está orientado para aquellas personas interesadas en el control del robot móvil Robotino de FESTO y en la utilización del algoritmo de búsqueda informada A*, utilizando el API para la comunicación y programación del Robotino bajo C# .NET, con el fin que resulte fácil su comprensión y aplicación en encontrar el camino más óptimo dentro de un entorno controlado. La información necesaria se encuentra estructurada de la siguiente forma:

En el primer capítulo se menciona de manera general una introducción, los objetivos, el alcance y el planteamiento del problema sobre el presente proyecto de titulación.

El capítulo 2 presenta una revisión sobre los robots móviles y la Inteligencia Artificial. En los robots móviles se hace una revisión sobre los tipos de robot móviles, sensores, actuadores, arquitecturas de control, tipos de entornos y sus diferentes aplicaciones, y en la Inteligencia Artificial se describen las diferentes estrategias de búsqueda junto con algunos de sus algoritmos de búsqueda.

En el capítulo 3 se da un análisis y diseño del funcionamiento del sistema de búsqueda, como también se da una breve revisión de cada uno de los componentes del robot móvil Robotino de FESTO.

Y finalmente en el capítulo 4 se presenta un informe de las pruebas realizadas, los resultados obtenidos y sus respectivas conclusiones y recomendaciones que se dieron durante el trabajo del presente proyecto de titulación.

CAPÍTULO 1

1.1 INTRODUCCIÓN

La robótica móvil en la actualidad está siendo empleadas en diferentes aplicaciones de investigación y exploración, realizando tareas peligrosas, desagradables o tediosas para el ser humano, y junto con la Inteligencia Artificial dotan al robot de mayor autonomía, es decir, una menor supervisión humana en las tareas de planificación, percepción y control para el cumplimiento de los objetivos.

En el presente proyecto de tesis se ha implementado un sistema de búsqueda informada que hace uso del algoritmo A* de Inteligencia Artificial para encontrar el camino más óptimo dentro de un entorno controlado utilizando el robot móvil Robotino de FESTO en tiempo real.

En cuanto al control del Robotino se empleó la arquitectura deliberativa¹, ya que el sistema de búsqueda está constituido por una parte de hardware y otra de software. La parte de hardware está formada por el robot móvil que opera bajo un sistema operativo GNU/Linux, que a su vez se comunican a través de una red inalámbrica WLAN a un computador que opera bajo Windows XP donde se ejecuta la aplicación de búsqueda y el API que permite la comunicación entre ellos.

La aplicación de búsqueda esta implementada en el lenguaje de programación C#.NET, donde el usuario tiene la posibilidad de crear y editar entornos virtuales formados de arcos y nodos², con el fin de observar y comprender de mejor manera el funcionamiento del algoritmo, utilizando el Robotino con algunas configuraciones adicionales o dentro de la misma aplicación.

¹ OLLERO BARTURONE, Aníbal, *ROBÓTICA – Manipuladores y robots móviles*, 1^{ra}. Edición, Editorial Marcombo, Barcelona-España, 200, p. 152

² RUSSELL, Stuart J. y NORVING Peter, *Inteligencia Artificial Un Enfoque Moderno*, 2^{da}. Edición, Editorial Pearson Educación S.A., Madrid-España, 2004, p.70, 80.

1.2 OBJETIVO GENERAL

Implementar un algoritmo de búsqueda informada en el robot móvil Robotino de FESTO para encontrar el camino con la solución más óptima dentro de un entorno controlado.

1.2.1 OBJETIVOS ESPECÍFICOS

- Conocer el manejo y funcionamiento de los componentes que posee el robot móvil utilizando la documentación disponible para el mismo.
- Utilizar el API desarrollado para el Robotino de FESTO bajo C#.NET desde una PC.
- Seleccionar los componentes más apropiados del robot móvil para guiarlo por los caminos más óptimos dentro del entorno de trabajo.
- Definir la heurística apropiada en el algoritmo de búsqueda informada para guiar el proceso de búsqueda por los caminos más óptimos hacia el objetivo.
- Utilizar una arquitectura de control que permita al robot móvil cumplir las tareas de manera organizada.
- Desarrollar un software que muestre los resultados obtenidos del proceso de búsqueda, ya sea utilizando al Robotino de FESTO dentro del entorno de trabajo o en el entorno virtual para observar los caminos más óptimos generados por el algoritmo de búsqueda informada.

1.3 ALCANCE DEL PROYECTO

Para el desarrollo de este proyecto de titulación se utilizará el robot móvil Robotino de FESTO adquirido por la universidad para los laboratorios de Electrónica, haciendo uso de sus componentes de hardware y del API desarrollado para el control del robot móvil.

El robot móvil se desplazará por los caminos generados por el algoritmo de búsqueda informada desde una posición inicial (nodo inicio) hasta llegar a su objetivo final (nodo objetivo), encontrando el camino más óptimo.

El desplazamiento del robot móvil será filoguiado, usando un solo sensor inductivo que permitirá la detección y seguimiento de los caminos generados por el algoritmo de búsqueda informada dentro del entorno de trabajo.

El algoritmo de búsqueda informada para el robot móvil será desarrollado bajo la plataforma .NET, utilizando el lenguaje de programación C#.

El robot móvil Robotino de FESTO solo funcionará dentro del entorno planteado para desplazarse por los caminos más óptimos.

1.4 PLANTEAMIENTO DEL PROBLEMA

La mayoría de aplicaciones desarrolladas para el robot móvil Robotino de FESTO se limitan a tareas, como:

- Seguimiento de objetos y líneas empleando una cámara Web.
- Detección y evasión de obstáculos utilizando sus sensores.
- Detección de distancias mediante sensores de luz infrarroja.

A pesar de realizar todas las tareas mencionadas anteriormente de forma autónoma, las capacidades explotadas son mínimas por no tener un mayor grado de autonomía que le permita al robot móvil tomar decisiones para encontrar el camino más óptimo dentro de un entorno real.

CAPÍTULO 2

MARCO TEÓRICO

2.1 ROBOTS MÓVILES

Los robots móviles son máquinas que poseen sistemas electromecánicos³, que utilizan técnicas de navegación automática, con el fin de incrementar las aplicaciones en la robótica móvil cumpliendo tareas de percepción, planificación y control por sí mismo en entornos reales con una limitada intervención humana.

En los años sesenta se comienza con el estudio de los robots móviles autónomos aplicados principalmente en la industria, siendo capaces de desplazarse siguiendo un cable enterrado o detectar marcas por el piso utilizando sus sensores ópticos, a estos robots móviles se les conoce también como vehículos filoguiados.

En los años 70 se logra incrementar la autonomía en los robots móviles obteniendo mayor control. Sin embargo, aquella tecnología de la época fue insuficiente para desarrollar una navegación eficiente. Muchas de las aplicaciones se orientaban a sistemas de visión, aunque no se tuvo mucho éxito por la capacidad computacional y el requerimiento de sensores más eficientes.

El mundo de los robots móviles autónomos empieza a partir de los años 80, siendo capaces de realizar tareas de planificación, percepción y control en pequeños intervalos de tiempo, y con el desarrollo de nuevas tecnologías como: Computadoras con mayor procesamiento de información, sensores y mecanismos mucho más eficientes permitían a los robots móviles detectar obstáculos, tomar decisiones, etc.

³ Sistemas electromecánicos.- son los que combinan partes eléctricas y mecánicas para conformar un mecanismo.

Con los avances en la computación se logra implementar técnicas de Inteligencia Artificial en los robots móviles para incrementar su grado de autonomía.

En cuanto a los robots móviles autónomos son todos aquellos que perciben su entorno por medio de sus sensores, y actúan sobre el mismo sin o con poca intervención humana, tales como: Seguidor de líneas en pistas con diferentes desniveles, buscador de luminosidad, etc. Se considera que un robot móvil completamente autónomo es capaz de recibir información de su entorno, desplazarse desde un estado inicial a un estado final, evitar obstáculos sin la intervención humana, y también pueden tener la capacidad de tomar decisiones y ajustarse a nuevas estrategias para cumplir los objetivos en beneficio del usuario.

Uno de los principales problemas que se tiene en los robots móviles se encuentra en su sistema de navegación, en la actualidad se están utilizando algoritmos computacionales y sistemas GPS⁴ para su posicionamiento y orientación, basados en técnicas de Inteligencia Artificial, reduciendo los errores acumulados por técnicas de odometría y sensores tales como: Acelerómetros, giróscopos, etc. [Ollero Aníbal 2001]

2.1.1 SISTEMAS DE LOCOMOCIÓN PARA ROBOTS MÓVILES

De acuerdo al entorno en el que interactúan los robots móviles, se han desarrollado diferentes medios de locomoción (Oruga, patas, ruedas, etc.) para desplazarse por un determinado medio.

⁴ Global Positioning System- (Sistema de Posicionamiento Global)

Locomoción por orugas o pista de deslizamiento

Se desplazan con gran maniobrabilidad por diferentes terrenos. La desventaja de utilizar estos medios de locomoción es su gran consumo de energía para conseguir mayor tracción al momento de desplazarse o realizar giros.



Figura 2.1: Irobot 710 Warrior (iROBOT)
Fuente: [Locomoción por orugas]

Locomoción por patas

Pueden llegar a ser muy complejos dependiendo del número de patas que posea, entre estos se han desarrollado bípedos (se asemejan al humano), cuadrúpedos y hexápodos que tienen dificultades en su equilibrio, razón por la cual el control de sus articulaciones no es tan sencillo, obteniendo bajas velocidades al momento de desplazarse de un lugar a otro.

Ejemplos:



Figura 2.2: Asimo – robot humanoide (HONDA)
Fuente: [Robot Bípedo]



Figura 2.3: Aibo – es un robot mascota (SONY)
Fuente: [Robot cuadrúpedo]



Figura 2.4: Robot hexápodo Silo6 (Department of Automatic Control, Industrial Automation Institute and Institute Spanish National Research Council)
Fuente: [Robot hexápodo – Silo6]



Figura 2.5: The ATHLETE Rover (NASA)
Fuente: [Robot hexápodo – Athlete Rover]

Locomoción por ruedas

Este tipo de locomoción ha tenido mayor acogida por ser fácil de construir, consumir una menor cantidad de energía y alcanzar velocidades relativamente altas en sus desplazamientos con ruedas de algún tipo.



Figura 2.6: Robot Nomad desarrollado por investigadores del departamento de robótica de la Universidad Carnegie Mellon, con financiación de la NASA.

Fuente: [Robot con ruedas]

2.1.2 CONFIGURACIÓN CINEMÁTICA DE LOS ROBOTS MÓVILES CON RUEDAS

Existen diferentes configuraciones para los sistemas de locomoción con ruedas de acuerdo al uso que se le quiera dar. Entre estas configuraciones encontramos:

2.1.2.1 CONFIGURACIÓN ACKERMAN

Son sistemas muy estables utilizados por los automóviles, es decir, constituidos por cuatro ruedas convencionales, dos de ellas son de dirección ubicada normalmente en la parte delantera, y dos en la parte trasera que proporcionan tracción incluso en terrenos inclinados.

Las ruedas de dirección en esta configuración tienen una relación entre sus ángulos para eliminar el deslizamiento. La relación entre los ángulos de dirección se muestra en la siguiente ecuación de Ackerman:

$$\cot\theta_i - \cot\theta_o = \frac{d}{l}$$

Dónde:

θ_i Ángulo relativo a la rueda interior

- θ_o Ángulo relativo a la rueda exterior
- d Separación lateral entre ruedas
- l Separación longitudinal entre ruedas

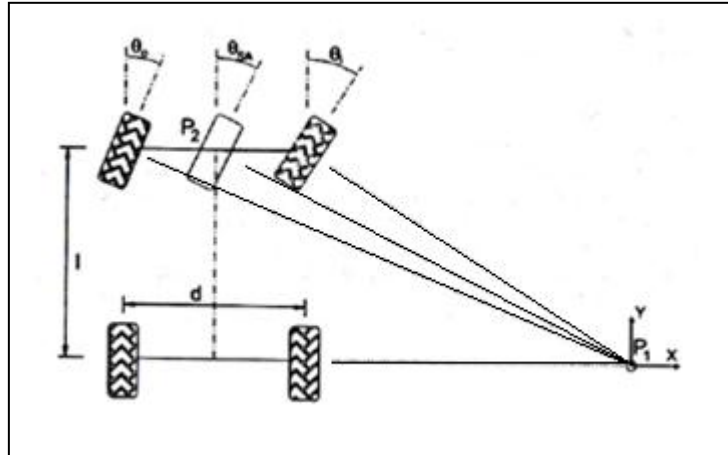


Figura 2.7: Posición de las ruedas en configuración Ackerman
Fuente: [Robots móviles]

Uno de los inconvenientes de esta configuración viene dado por el radio de giro, ya que no es muy pequeño, razón por la cual debe disminuir la velocidad al entrar a una curva.

2.1.2.2 CONFIGURACIÓN TRICICLO

Esta configuración posee tres ruedas convencionales en forma triangular similar al triciclo de los niños. La rueda delantera se utiliza tanto de dirección y tracción mientras que las ruedas traseras se mueven libremente. Asimismo, se tienen problemas de estabilidad al maniobrar por terrenos inclinados ocasionando pérdida de tracción, ya que la rueda de tracción tiende a elevarse perdiendo contacto con el terreno.

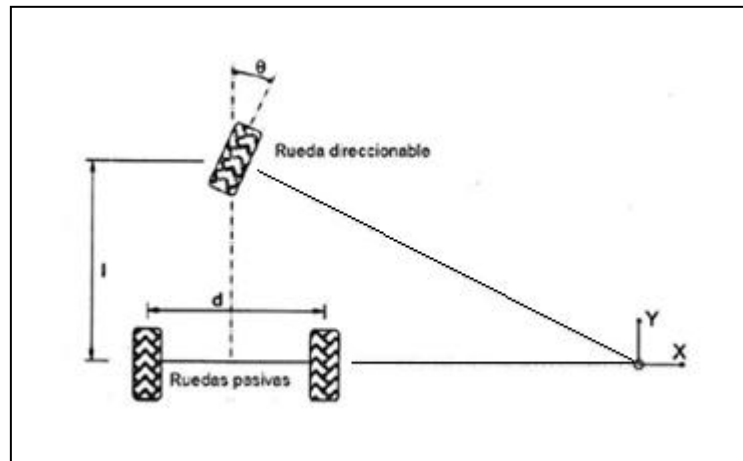


Figura 2.8: Configuración triciclo
Fuente: [Ollero Aníbal 2001]

2.1.2.3 CONFIGURACIÓN DIFERENCIAL

Son fáciles de construir por tener dos ruedas convencionales de tracción independientes, las mismas que generan su direccionamiento empleando la diferencia de velocidades. Utilizan una o más ruedas de apoyo (ruedas de castor, ruedas de bolas, ruedas suecas) según la necesidad para evitar la inclinación del vehículo. Sin embargo, la presencia de tres o más ruedas de apoyo ocasionaría pérdidas de tracción al navegar por terrenos irregulares, quedando suspendidas en el aire por las ruedas de apoyo.

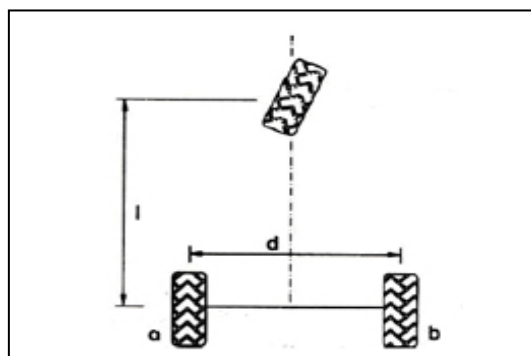


Figura 2.9: Disposición de las ruedas en configuración diferencial con una rueda loca al frente
Fuente: [Robots móviles]

2.1.2.4 CONFIGURACIÓN SÍNCRONA

Se trata de una configuración compleja en su diseño e implementación. Sin embargo, el control de sus movimientos es muy sencillo. Utiliza dos motores, cada motor mueve tres o más ruedas de algún tipo simultáneamente, empleando bandas dentadas para el desplazamiento y dirección.

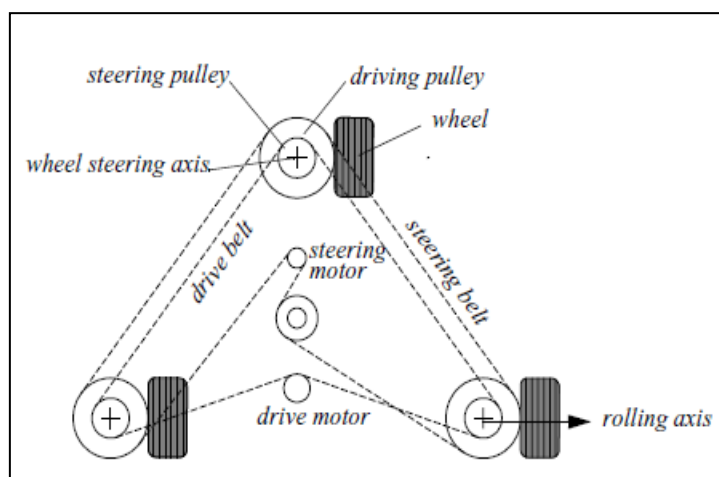


Figura 2.10: Configuración Síncrona con tres ruedas
Fuente: [Siegwart Roland 2004]

2.1.2.5 CONFIGURACIÓN OMNIDIRECCIONAL

A diferencia de las demás configuraciones permite una excelente maniobrabilidad en los robots móviles, y un desplazamiento sin la necesidad de una orientación específica usando ruedas suecas. Normalmente esta configuración requiere más de dos ruedas que proporcionan tracción y dirección a la vez, lo que le permite al robot móvil realizar movimientos complicados. Sin embargo, su complejo control no asegura un desplazamiento en línea recta.

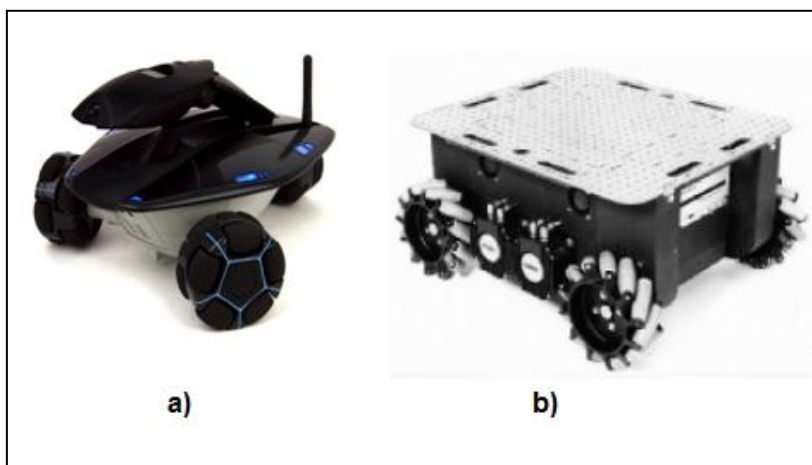


Figura 2.11: a) Robot omnidireccional Rovio (WowWee), b) Uranu mobile robot (Carnegie- Mellon University).

Fuente: [Rovio], [Siegwart Roland 2004]

Tipos de ruedas

	
De Bolas	Castor
	
Omnidireccionales	Convencionales

Figura 2.12: Tipos de ruedas para robots móviles

Fuente: [Tipos de ruedas]

2.2 PERCEPCIÓN

La percepción permite a los robots móviles autónomos adquirir la información necesaria sobre su entorno de trabajo en cada instante, y de esta manera poder interactuar con el medio que lo rodea cumpliendo los propósitos para las que fue diseñado. Esta tarea se realiza mediante el uso de los diferentes sensores.

Los sensores son dispositivos que en su mayoría utilizan procesamiento electrónico, que consiste en la interpretación de información y evaluación por parte del usuario final para el control de robots móviles, a través de ordenadores u otro mecanismo como los microcontroladores.

La información adquirida por parte de los sensores es transformada de una forma de energía a otra, normalmente a valores de salida analógicas y digitales, por ejemplo: Los fototransistores son elementos sensibles a la luz que trabajan al igual que los transistores normales, con la diferencia que estas se activan con mayor cantidad de luz reflejada y se desactivan con la ausencia de ella, esta información adquirida en el mundo digital se lo denomina código binario (0 y 1), facilitando un mayor entendimiento en el control electrónico de robots móviles y manipulación por parte del usuario final.

Los sensores juegan un papel muy importante en la navegación de un vehículo filoguiado, proporcionando información de la velocidad, posición, orientación del vehículo respecto al entorno de trabajo, información que es utilizada para seguir una trayectoria o generar una trayectoria basada en mapas, permitiendo al robot móvil realizar una navegación segura libre de obstáculos.

A continuación se mencionan los diferentes sensores utilizados en los robots móviles para realizar tareas de percepción del entorno.

2.2.1 MICROINTERRUPTORES DE CHOQUE O BUMPERS

Un Bumper es un conmutador de dos posiciones con un muelle de retorno, este muelle permite a la palanca de accionamiento regresar a su estado de reposo una vez que fue accionada.

Es muy común ver este sensor colocado en los vehículos filoguiados para la detección de obstáculos por contacto directo, este dispositivo no es muy útil cuando el móvil alcanza altas velocidades, puesto que el móvil no tiene el suficiente tiempo para detenerse.



Figura 2.13: Microinterruptor de choque o bumper
Fuente: [Sensores]

2.2.2 SENSORES DE PRESENCIA Y PROXIMIDAD

Son dispositivos electrónicos que en la robótica móvil permiten detectar la presencia de un objeto y la distancia a la que este se encuentre. Algunos de los sensores son mucho más sofisticados, capaces de enviar y recibir información, tales como los sensores de ultrasonido o medidores de distancia.

A parte de los sensores de ultrasonido y dispositivos mecánicos (fines de carrera) que generan señales binarias, existen diferentes métodos para detectar obstáculos tales como, el uso de visión artificial utilizando librerías, tales como Opencv⁵, una librería de tratamiento de imágenes destinada principalmente a aplicaciones de visión

⁵ Open Source Computer Vision Library

por computador en “tiempo real”⁶, siendo estas aplicaciones programables en diferentes lenguajes de programación, logrando realizar en los robots móviles tareas de percepción mucho más avanzadas.

A estos sensores también se los denomina como sensores externos, que permiten al robot móvil interactuar con su entorno, cumpliendo tareas definidas por el usuario final, ya que en muchas de las aplicaciones desarrolladas para el control de robots móviles están ubicados en partes visibles. El uso de varios de estos sensores en los robots móviles aumenta la complejidad en el control de los mismos y el grado de autonomía que posea el robot será mucho mayor.

2.2.3 SENSORES INDUCTIVOS

Son aquellos que sirven para detectar objetos metálicos sin la necesidad de un contacto físico. Su funcionamiento consiste en detectar un objeto metálico al aproximarse a la distancia de sensado, donde se producen pérdidas de corriente o corrientes de Foucault [Pallás Ramón, 2005, p.192], ocasionando variaciones en el campo magnético por las propiedades del material. Estas variaciones son interpretadas por el circuito de disparo que monitorea la amplitud del oscilador, que a su vez, pueden ser bruscas o suaves cuando el objeto se encuentra en el área de sensado, generando una salida a un umbral predeterminado de “ON” y “OFF”.

Los diferentes valores de salida generados por un sensor inductivo pueden variar de acuerdo a la distancia del objeto y al tipo de material.

⁶ Tiempo real: intervalo de tiempo que podemos esperar entre la entrada y la salida de una señal.

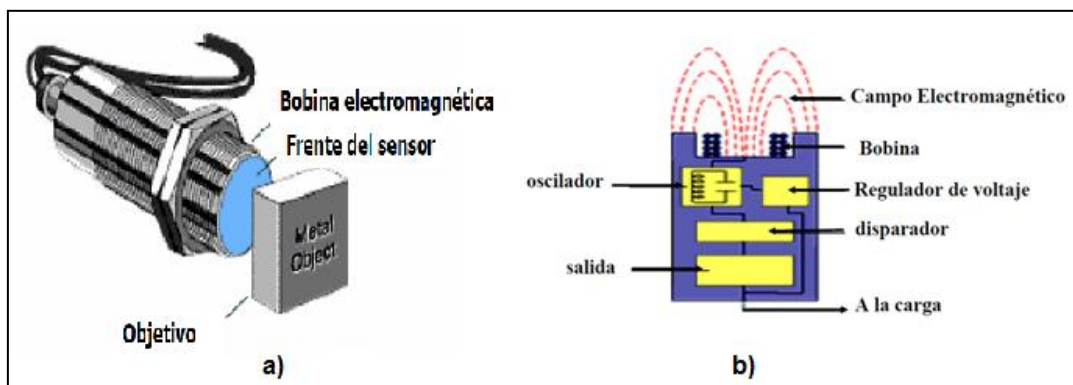


Figura 2.14: a) Sensor Inductivo, b) Componentes de un sensor Inductivo

Fuente: [Sensores Inductivos]

En la actualidad se pueden encontrar dos tipos de sensores inductivos, los blindados y no blindados. Los blindados proporcionan mayor resistencia a los golpes, aunque son limitados en el área de detección por su blindaje metálico, mientras que los no blindados proporcionan un área de sensado mucho mayor, pero, estos son poco resistentes.

Los sensores inductivos también tienen una amplia acogida en las industrias, en aplicaciones tales como: detectar la presencia y paso de piezas, rotación, contaje, etc., por ser inmunes al ruido y a cambios ambientales.

2.2.4 SENSORES ÓPTICOS

Son utilizados como elementos de seguridad para detectar pequeños o grandes movimientos. Generalmente son dispositivos de emisión y recepción de una haz infrarrojo, además, generan una salida binaria cuando se recibe una intensidad de luz superior a un umbral preestablecido.

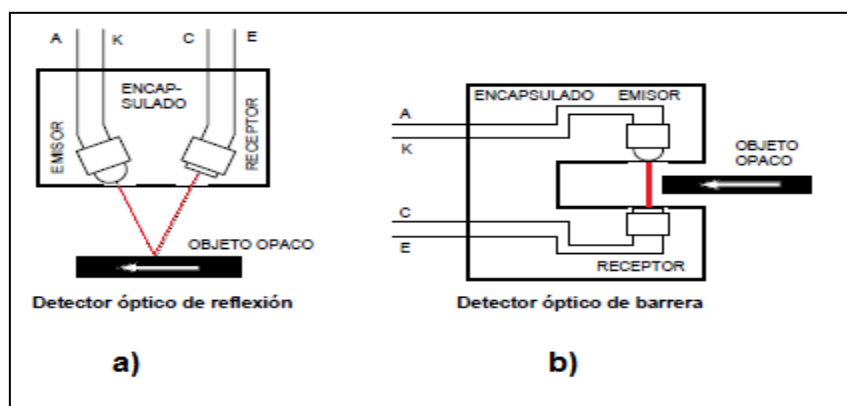


Figura 2.15: a) Detector óptico de reflexión, b) Detector óptico de barrera
Fuente: [García Juan 2007]

2.2.5 ENCODERS

Los encoders o codificadores ópticos son empleados para determinar la velocidad, posición y aceleración del robot móvil, su funcionamiento está basado en la interrupción continua de un haz de luz que percibe el fotodetector durante el giro del disco con secciones transparentes y opacas, indicando de esta manera el sentido de giro de las ruedas del robot móvil.

2.2.5.1 ENCODERS INCREMENTALES

Los encoders incrementales constan de un disco transparente con una serie de marcas opacas colocadas radialmente y equidistantes entre sí. Estos encoders poseen un elemento emisor de luz y un elemento fotoreceptor para determinar la posición y distancia recorrida de las ruedas del robot móvil. La figura 2.16 muestra los componentes de un encoder incremental.

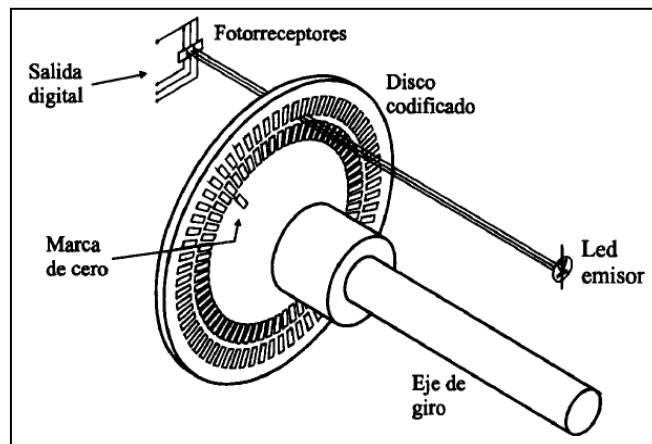


Figura 2.16: *Encoder Incremental*
Fuente: [Sensores Internos]

2.3 ACTUADORES

Los actuadores son aquellos que dotan de movimientos a un robot móvil utilizando tecnologías tanto hidráulicas, neumáticas y eléctricas. Estas tecnologías en la actualidad son investigadas para conseguir mayor precisión en muchas aplicaciones industriales.

En cuanto a las tecnologías mencionadas se definen de la siguiente manera:

Actuadores hidráulicos

Son utilizados cuando se requiere manejar altas potencias en base a fluidos a presión. Estos actuadores consumen mucha energía y requieren de un mantenimiento constante.

Actuadores neumáticos

Estos actuadores utilizan aire comprimido para realizar trabajos mecánicos, alcanzando altas velocidades pero a la vez limitadas por el escape de aire existente.

Actuadores eléctricos

Son los más utilizados en las industrias por ser de fácil manejo e instalación, puesto que requieren tan solo energía eléctrica para su funcionamiento.

Dentro de estos actuadores eléctricos encontramos motores de corriente continua (DC), motores de corriente alterna (AC) y motores de paso.

Motores de corriente continua (DC)

Son generalmente utilizados en la robótica móvil, por su bajo consumo de energía y fácil manejo cuando se requiere cambiar el sentido del giro del motor. Esto se consigue invirtiendo la alimentación de energía, y con el uso de reductores de velocidad se logra tener un mayor control de los mismos.

Motores de corriente alterna (AC)

Estos motores a diferencia de los de corriente continua (DC) requieren de mayor energía para su funcionamiento, siendo útiles en muchas aplicaciones dentro de la industria por su buen rendimiento y bajo mantenimiento.

Motores de paso

Un motor de paso es útil en aplicaciones cuando se requiere movimientos de mayor precisión. A diferencia de los motores de corriente continua (DC) y corriente alterna (AC), realizan giros controlados desde 1.8° hasta 360° , permitiendo al eje del motor enclavarse en una posición específica.

2.4 ARQUITECTURA PARA EL CONTROL DE ROBOTS MÓVILES

La arquitectura en los robots móviles se define como un conjunto de módulos de software y hardware que están interconectados entre sí, percibiendo, tomando decisiones y realizando acciones en tiempo real para cumplir una determinada tarea. Las arquitecturas permiten que las operaciones de control en los robots móviles se lleven de una manera organizada, obteniendo información de su entorno por medio de sus sensores y realizando acciones al mismo tiempo de forma autónoma.

2.4.1 CONTROL INTELIGENTE

Esta dado por el uso de funciones autónomas integradas en una arquitectura de control “[...] que permiten la realización parcial o totalmente autónoma de operaciones tales como la planificación de tareas, planificación de movimientos, percepción sensorial y la reacción ante la presencia de obstáculos y condiciones no previstas en general”⁷. En algunos casos es necesario utilizar las funciones teleoperadas y autónomas en conjunto, no obstante, su control será mucho más complejo.

2.4.2 REQUERIMIENTOS GENERALES DE LA ARQUITECTURA

Al momento de diseñar una arquitectura se deben tomar en cuenta requerimientos, tales como:

Programabilidad

Capacidad de los robots móviles para realizar múltiples tareas dentro de un entorno, donde su arquitectura de control debe cumplir objetivos realizando diferentes tipos de acciones, por ejemplo, realizar secuencia de movimientos para la generación de

⁷ OLLERO, Aníbal. Op. Cit. p. 146.

trayectorias, toma de información útil por parte de los sensores, evasión de obstáculos, etc.

Eficiencia

Es el resultado obtenido por los robots móviles al momento de realizar una tarea, tomando en cuenta factores como la precisión, tiempo de ejecución y recursos de Hardware y Software empleados para el cumplimiento de sus objetivos, es decir: que tan bien soluciona el problema planteado.

Capacidad de evolución

Se define a la integración de tecnologías modernas en los robots móviles, tales como: tarjeta de adquisición de datos, GPS para posicionamiento y orientación, proporcionando mayor control y autonomía en ellos.

Grado de autonomía

Es cuanto intervienen las personas sobre el robot móvil para la ejecución de sus tareas, mientras menos intervención se tenga mayor será su grado de autonomía. Esto normalmente se pueden observar en arquitecturas de control inteligente, ya que estas poseen complejos controles para la ejecución de sus tareas.

Fiabilidad

Es la no dependencia de un solo sistema de control en los robots móviles, utilizados para incrementar la seguridad de su funcionamiento, redundando operaciones de control.

Por ejemplo, la fiabilidad se observa cuando se emplea un sistema de visión artificial y un sistema basado en sensores de proximidad para la detección de obstáculos.

Ambos sistemas proporcionan la misma información de diferente manera, pero el mal funcionamiento de uno de ellos no impediría el cumplimiento de sus objetivos.

Adaptabilidad

Capacidad que posee un robot móvil autónomo para modificar sus comportamientos en entornos poco conocidos y realizar un desempeño óptimo. En esta parte el robot móvil analiza la información proporcionada por los sensores y reacciona rápidamente ante posibles eventualidades, realizando acciones oportunas para el cumplimiento de las tareas.

2.4.3 TIPOS DE ARQUITECTURAS DE CONTROL

Se tienen diferentes tipos de arquitecturas que son empleadas en los robots móviles tomando en cuenta su eficiencia, tiempo de respuesta y capacidad de aprendizaje.

Entre las arquitecturas de control para robots móviles se mencionan algunas como:

Arquitectura de control reactivo

Son útiles en los robots móviles cuando se requieren respuestas muy rápidas en la ejecución de sus tareas, obteniéndose comportamientos de percepción – acción sin la necesidad de realizar procesamientos complejos de la información de sus sensores, donde el sistema realiza una simple consulta a una colección de reglas para decidir qué acciones realizar. Esta arquitectura carece de precisión en los movimientos realizados, esto se puede observar claramente en un seguidor de línea.

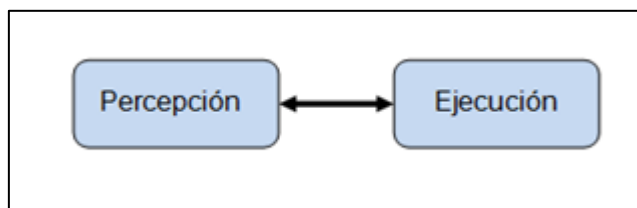


Figura 2.17: *Arquitectura Reactiva*
Fuente: Autores

Arquitectura de control deliberativo

Son utilizadas en la búsqueda y planificación de caminos más óptimos para cumplir sus objetivos, a diferencia de la arquitectura de control reactivo son más precisas en sus movimientos, realizando varias acciones desde un estado inicial al final usando métodos simbólicos de Inteligencia Artificial. Esta arquitectura está limitada en la capacidad de almacenamiento y tiempo de procesamiento de información para realizar una acción.

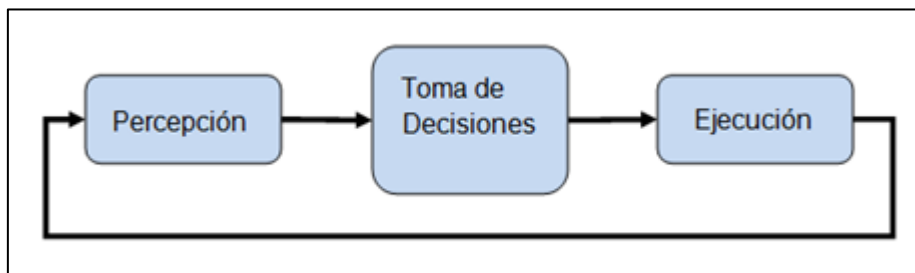


Figura 2.18: *Arquitectura Deliberativa*
Fuente: Autores

Arquitectura de control híbrido

Se basa en la cooperación entre la arquitectura reactiva y deliberativa. La arquitectura reactiva proporciona funciones de seguridad, mientras que la deliberativa se encarga de tomar decisiones para realizar acciones oportunas.

Estas arquitecturas trabajan paralelamente en un solo sistema de control dividido en varios subsistemas, realizando una comunicación entre ellos para obtener mayor fiabilidad en los robots móviles.

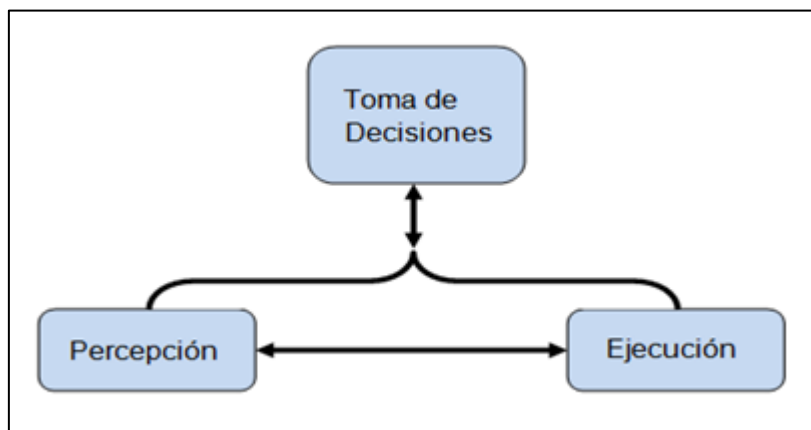


Figura 2.19: Arquitectura Híbrida
Fuente: Autores

2.4.4 TIPOS DE AMBIENTES

Describe el entorno donde el robot móvil autónomo realizará las diferentes acciones para cumplir los objetivos encomendados por el usuario. Existen varios tipos, algunos de ellos requieren de técnicas de Inteligencia Artificial.

Totalmente observable y parcialmente observable

Cuando el Robot móvil detecta todos los aspectos del entorno haciendo uso de sus sensores en cada instante para cumplir sus objetivos, se dice que el medio es totalmente observable, mientras que en un medio parcialmente observable no se tienen todas las características del entorno debido a la presencia de ruido, sensores poco exactos u otros factores que afectan en la toma de decisiones.

Determinista y estocástico

Se denomina estocástico cuando la acción del robot móvil funciona por el azar, incrementado de esta manera la incertidumbre al momento de ejecutar las tareas, y determinista cuando las acciones están predeterminadas

Episódico y secuencial

Un medio episódico divide la experiencia del Robot móvil en episodios, realizando una única acción en cada una de ellos, sin tomar en cuenta las decisiones anteriores, mientras el secuencial debe de tomar en cuenta las acciones realizadas previamente para tomar decisiones futuras.

Estático y dinámico

Se denomina medios estáticos a aquellos que no sufren ninguna modificación en su estructura cuando el Robot móvil realiza las diferentes acciones, de lo contrario se dice que es un medio dinámico.

Discreto y continuo

En un medio discreto las percepciones y acciones están claramente definidas, es decir, el robot móvil sabe exactamente en qué estado se encuentra durante la ejecución de sus tareas dentro del entorno para llegar a su objetivo, de no ser así, se le considera un medio continuo.

2.4.5 NAVEGACIÓN EN ROBOTS MÓVILES

Son técnicas que permiten llevar al robot móvil desde un punto inicial a un punto final con una serie de desplazamientos, reaccionando ante situaciones inesperadas de

forma segura a medida que esta atraviesa un entorno evitando obstáculos fijos y móviles sin perderse.

Las tareas involucradas en la navegación son: percepción del entorno a través de sus sensores, planificación de trayectoria libre de obstáculos o caminos construidos a base del modelo del entorno y guiados de robots móviles mediante marcas de referencia.

A parte de la navegación se deben tomar en cuenta conceptos fundamentales como la operación y misión en los robots móviles. La operación en los robots móviles se refiere a la interacción con objetos de su entorno de trabajo para cumplir ciertos objetivos especificados, aumentando en algunos casos su complejidad en entornos no estructurados. La misión es la cooperación de la navegación y operación para cumplir una serie de objetivos sin que el robot móvil colisione con los objetos del entorno, la figura siguiente muestra un esquema básico de arquitectura para realizar una misión.

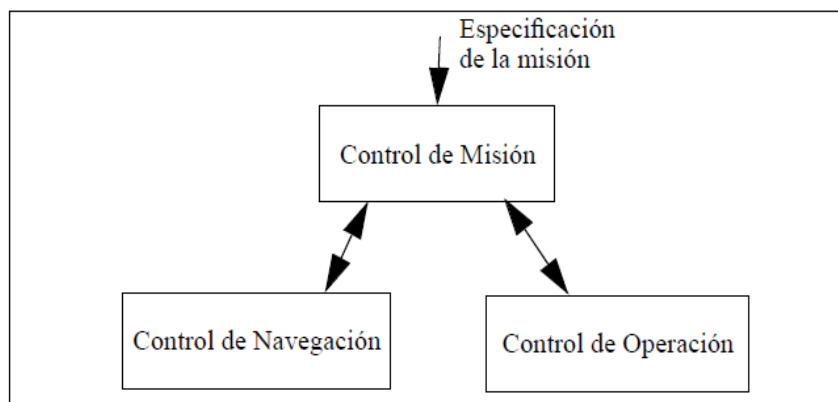


Figura 2.20: Esquema básico de la arquitectura necesaria en un robot móvil para realizar una misión.
Fuente: [Ollero Aníbal 2001]

La arquitectura muestra un control de misión que establece una coordinación entre el control de navegación y el control operación que deben tener un perfecto funcionamiento para cumplir las especificaciones de la misión asignada por el usuario.

2.4.6 ESQUEMA DE NAVEGACIÓN EN UN ROBOT MÓVIL

Son un conjunto de tareas básicas empleadas para construir un camino libre de obstáculos en entornos estructurados o no estructurados y para que un robot móvil pueda adaptarse en cualquier entorno se debe utilizar la planificación global y local, haciendo el uso diferenciado de cada una de ellas al momento de la navegación.

Planificación global

El móvil realiza una aproximación del camino que lo llevará a la meta, utilizando el control de misión para resolver los problemas encontrados en cada submeta.

Planificación local

Se lleva a cabo en tiempo de ejecución para evitar colisiones con objetos que no se tomaron en cuenta en la planificación global, determinando el camino real que debe seguir el robot móvil. Su planificación está basada en la información proporcionada de sus sensores externos.

Cuando los entornos son perfectamente estructurados se requiere el uso tan solo de la planificación global para encontrar el camino final, de lo contrario se requiere un uso intensivo de la planificación local.

La siguiente figura 2.21 muestra un esquema de navegación básico para robots móviles en entornos de trabajo libre de obstáculos.

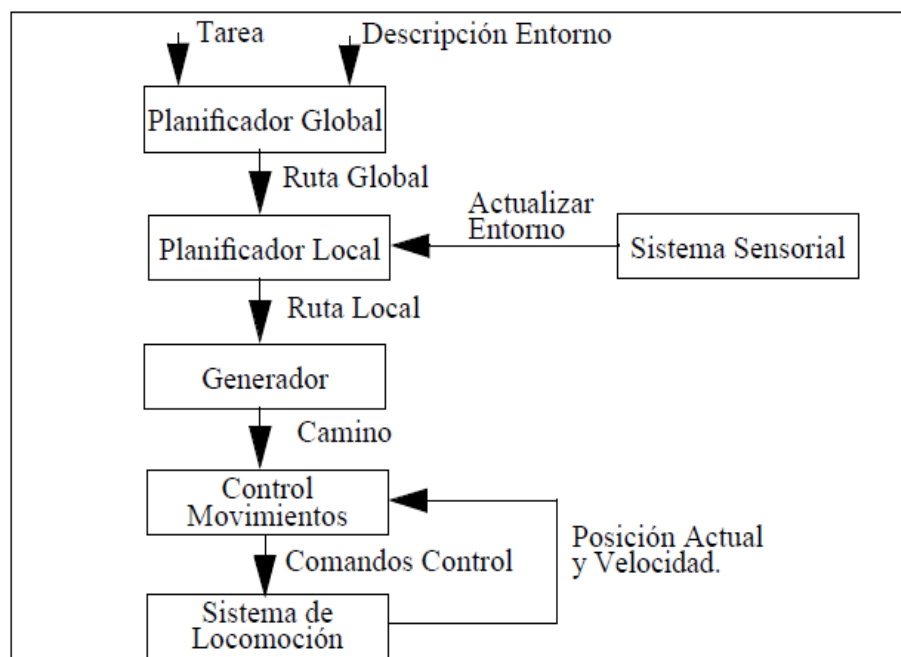


Figura 2.21: Navegador implantado en el robot móvil Blanche de AT&T

Fuente: [Ollero Aníbal 2001]

El esquema presentado hace uso de la planificación global y local para encontrar el camino hacia la meta final en forma continua, enviando información de la ruta a la entrada del generador para el control de sus movimientos obviando características cinemáticas y dinámicas⁸ del robot móvil. Su uso principalmente se da en entornos poco conocidos.

Planificación de la ruta

Es encontrar un camino seguro y libre de obstáculos desde la posición actual en la que se encuentra el robot móvil, hasta llegar a la meta ordenada por el nivel superior de la arquitectura de control.

⁸ Cinemática: estudio del movimiento del móvil sin considerar las fuerzas que lo colisionan.
Dinámica: estudio de las fuerzas que se requieren para ocasionar el movimiento del móvil.

Generador de caminos

Consiste en construir una sucesión de configuraciones que permiten al robot móvil la ejecución de un camino con el menor error posible al momento de realizar la tarea de navegación desde una posición inicial a la final, convirtiendo la ruta en un camino donde el robot móvil no necesariamente debe ser omnidireccional para un seguimiento de camino⁹ adecuado.

2.4.7 APLICACIONES

En la actualidad existen numerosas aplicaciones para los robots móviles, su diseño y construcción se basa de acuerdo a las necesidades del hombre para resolver diferentes problemas, ya sea de exploración, doméstico, educativo, aplicaciones especiales, etc. En su mayoría estas aplicaciones poseen sistemas de control autónomos que funcionan con poca intervención humana.

Transporte

Es utilizada principalmente en las industrias para trasladar cargas de diferentes volúmenes, siendo estos en su mayoría guiados de un lugar a otro a través de caminos fijos por marcas o metales situados en el piso. El seguimiento de estos caminos se realiza mediante sensores ópticos, inductivos, luz, etc.

Por motivos de seguridad los robots móviles filoguiados se dotan de sensores de proximidad para evitar posibles colisiones.

⁹ Camino se describe a un conjunto de puntos elegidos de la ruta que el móvil debe seguir para llegar a la meta.



Figura 2.22: Agv Eagle E210 Flexcart (CORECON)

Fuente: [Vehículos de guiado automático]

Robot de limpieza

Las tareas de limpieza son tediosas y repetitivas, razón por la cual en los últimos años se han desarrollado robots autónomos de limpieza, que son capaces de navegar por una habitación o por toda la casa, evitando obstáculos mediante sus sensores, al tiempo que aspiran el polvo y las pelusas que encuentra en el suelo. Estos robots son capaces de dirigirse automáticamente a una estación base para recargar sus baterías y seguir trabajando.



Figura 2.23: a) Robot de limpieza del hogar Roomba 531 (iRobot), b) Robot de limpieza de piscinas Solar-Breeze (Eco Pool Technologies)

Fuente: [Robot de limpieza del hogar], [Robot de limpieza de piscinas]

Los robots de limpieza no solo se restringen a tareas dentro del hogar, también podemos encontrar robots para la inspección y limpieza de ductos de aire acondicionado y ventilación en las industrias.



Figura 2.24: Robot Trenz-Clenear (Trenz-Clenear)
Fuente: [Robot de limpieza industrial]

Robot de vigilancia y seguridad

Estos robots de vigilancia son capaces de realizar múltiples tareas, como rondas en exteriores, comprobar perímetros, detectar y seguir a intrusos activando una alarma en el centro de control de forma remota. Todas estas tareas son realizadas eficazmente, incluso pueden trabajar en condiciones de oscuridad, humo, polvo o niebla gracias a su sistema de detección de intrusos.



Figura 2.25: Robot mSecurit (MoviRobotics)
Fuente: [Robot de vigilancia]

Robot de ayuda

Son Robots móviles utilizados en su mayoría para personas con discapacidades físicas, construidas con alguna tecnología que le sirva como ayuda técnica para realizar diferentes actividades, tales como: Dar de comer a los pacientes, manipular objetos, desplazarse de una forma autónoma, etc. Esto permite a las personas que tienen algún impedimento físico valerse plenamente por sí mismas reduciendo su discapacidad.



Figura 2.26: Yurina – Robot asistencial japonés (Toyota)
Fuente: [Robot de asistencia]

En la figura 2.26 se aprecia a Yurina un Robot asistencial japonés presentado en la feria Robotech Japan 2010 donde demostró su fuerza y habilidad para levantar y trasladar pacientes. Sus capacidades pueden llegar a acomodar a un paciente en la cama o llevarlo al baño con el mayor cuidado. Otra de las características de este robot es que puede convertirse en andador de rehabilitación o en silla de ruedas.

2.4.8 APLICACIONES ESPECIALES

En estas encontramos los robots de uso militar que tienen como misión recopilar información en lugares inexplorables y peligrosos para el hombre. Algunas de las

aplicaciones se encuentran aún en investigaciones, tales como: Robot desactivador de bombas o apagador de incendios.



Figura 2.27: Robot militar R-Gator con kernelGNU/Linux (iRobot)

Figura: [Robot militar]

En la figura 2.27 se presenta un robot móvil autónomo desarrollado para fines militares por la empresa iRobot llamado R-Gator. El Robot puede cambiar de modo autónomo a teleoperado y actuar también como explorador, apuntador o guardián. Cuando trabaja de modo autónomo el vehículo realiza un seguimiento al escuadrón militar como si se tratara de un soldado más, mediante un dispositivo llevado en el líder.

Uso lúdico

Son Robots de entretenimiento o acompañamiento, usados por las personas como mascotas o juguetes dotados de distintas capacidades e inteligencia. La compañía Sony ha diseñado novedosos robots de entretenimiento como AIBO, la versión más actual posee un vocabulario en inglés y algunas palabras en español.



Figura 2.28: AIBO modelo ERS-7M3 (Sony)
Fuente: [Robot cuadrúpedo]

AIBO también puede responder y expresarse verbalmente con su propietario. Asimismo, puede realizar acciones y movimientos similares a las mascotas reales.

Existen otros desarrollos por parte de Sony como es QRIO. Un robot que puede caminar por dos pies, bailar energéticamente, conversar con las personas de una manera entretenida y responder en sus propias palabras. Debido a la información obtenida durante las conversaciones con el propietario, lo irá almacenado en la memoria como un recuerdo para realizar conversaciones mucho más naturales.

Robot de competencia

Son empleados en concursos dentro de varias instituciones donde los alumnos deben poner en práctica habilidades y conocimientos para resolver problemas usando diferentes tecnologías, donde se valoran aspectos básicos de la robótica como la interacción con el entorno, técnicas de planificación y controlabilidad.

En la actualidad existen varias pruebas de competencias para robots móviles entre las cuales podemos citar: sumo, laberinto, rastreadores, velocistas y Robocup como las más prestigiosas competencias internacionales.

Sumo

Dos robots móviles que luchan por sacar al oponente del tatami (área delimitada por una línea de color blanca o negra), donde el tiempo de reacción y potencia son fundamentales.

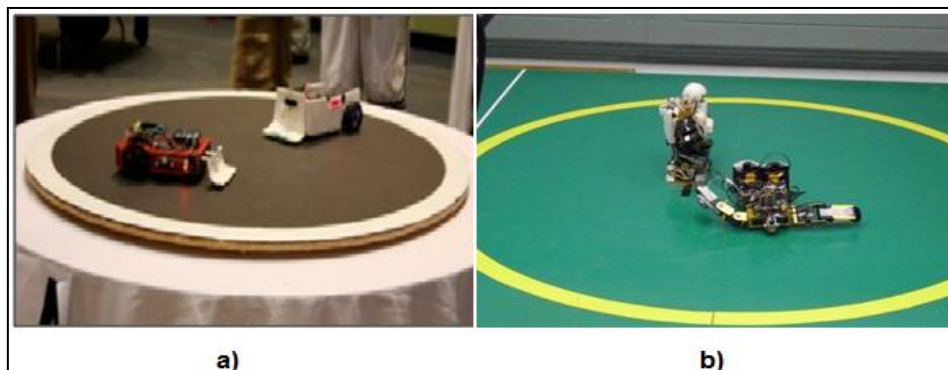


Figura 2.29: a) Robots con ruedas en combate de sumo, b) Robots humanoides en combate de sumo
Fuente: [Robots de competencia], [Luchadores de sumo Humanoides]

Laberinto

El robot móvil debe tomar decisiones y salir en el menor tiempo posible, utilizando algoritmos de Inteligencia Artificial o guiándose únicamente por la percepción de sus sensores.



Figura 2.30: Robot en una prueba de laberinto
Fuente: [Robots de competencia]

Velocista

Se toman en cuenta la velocidad y control al desplazarse por caminos sin fin marcados por una o dos líneas para determinar qué robot móvil es el más rápido. El robot móvil puede guiarse por una sola marca o ambas a la vez, siendo eliminado al salir del área delimitada por aquellas líneas.



Figura 2.31: Robot en prueba de velocidad
Fuente: [Robots de competencias]

Rastreadores

Son capaces de seguir caminos complicados llenos de curvas y bifurcaciones, donde el robot móvil debe llegar a la meta siguiendo el camino más corto en el menor tiempo posible.

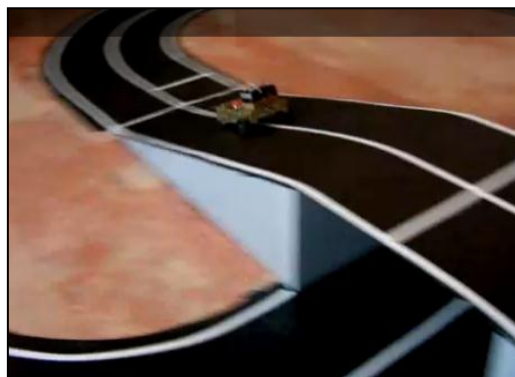


Figura 2.32: Robot rastreador
Fuente: [Robot rastreador]

Robocup

Consiste en jugar un partido de fútbol entre varios robots móviles autónomos que interactúan de forma cooperativa entre ellos y con su entorno.

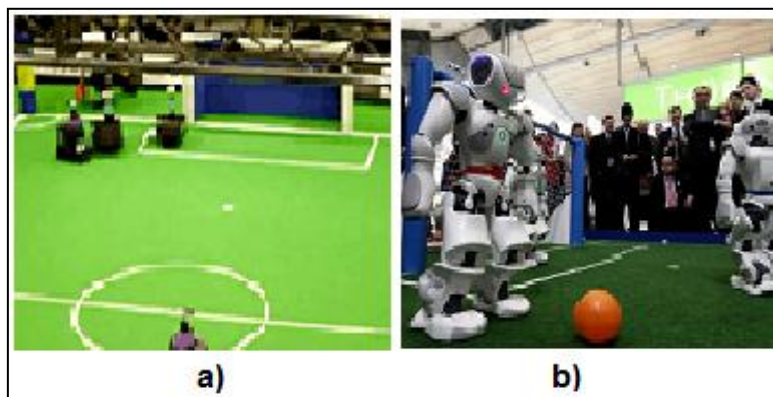


Figura 2.33: a) Robots de plataforma estándar b) Robots humanoides

Fuente: [Robocup]

2.5 INTELIGENCIA ARTIFICIAL

La Inteligencia Artificial (IA) es un campo nuevo con menos de 60 años de estudio, su concepto y desarrollo se originan a partir del siglo XX, donde, en el año de 1936 el genio matemático inglés Alang Turing diseña una máquina denominada “La Máquina de Turing”¹⁰ para comprobar si las máquinas pueden pensar al igual que un ser humano. Sin embargo, la respuesta ha quedado en una interrogante hasta el día de hoy, quedando a criterio de cada persona. Una de las mejores respuestas en nuestra perspectiva es que las máquinas son pensantes conforme incrementa su complejidad al resolver problemas. Por ejemplo, una máquina puede ser pensante al momento de interactuar con el entorno y tomar decisiones propias para el cumplimiento de tareas.

La Inteligencia Artificial tiene como objetivo comprender y reproducir el razonamiento humano, diseñando y construyendo máquinas capaces de realizar tareas iguales o

¹⁰ RUSSELL, Stuart J. y NORVING Peter, Op. Cit. p. 3.

mejores que los humanos. Mediante la implementación de programas inteligentes se intentan resolver problemas difíciles, analizando, organizando y convirtiendo los datos en conocimiento, de manera similar a como lo haría una persona, reproduciéndolo e implementándolo a través de herramientas de programación como: LISP, PROLOG, C, C++, C#, etc.

Los grandes avances de la computación con mayor capacidad de almacenamiento, mejor tiempo de procesamiento, etc., han permitido que la Inteligencia Artificial llegue a ser una metodología aplicada en hardware y software, que continuamente se sigue mejorando y modificando para desarrollar nuevas máquinas con mayor grado de inteligencia.

Actualmente la Inteligencia Artificial (IA) es aplicada en campos como las finanzas, la medicina (responder a diagnósticos médicos), la automatización y los juegos de estrategias (Ajedrez, Warcraft). Sin embargo, existen tareas fáciles de realizar para el hombre que son aún difíciles de realizar por las máquinas.

2.5.1 RESOLUCIÓN DE PROBLEMAS MEDIANTE BÚSQUEDAS

Las búsquedas tratan de encontrar la forma de resolver los problemas analizando sus posibles soluciones. En los inicios fueron aplicados en problemas simples (8-puzzle, tres en raya, apilar bloques, etc.) por las capacidades computacionales que se requerían. En la actualidad con los avances de la tecnología se logra incluir mucho más conocimiento y resolver problemas mucho más realistas, utilizando algoritmos de búsquedas sin información o heurísticos que reciben como dato de entrada un problema y generan una secuencia de acciones que conduzcan a la resolución.

Entre los problemas más conocidos tenemos: Los de juguetes (La aspiradora, 8-puzzle, problema de 8 reinas) que tienen una representación clara de la situación a resolver, y los del mundo real (Búsqueda de una ruta Turística, búsqueda en Internet,

viajante de comercio (PVC)) que tienen una representación mucho más compleja, ya que requieren de mayor información de la situación a resolver. Por ejemplo en un juego de 8-puzzle, la toma de decisiones es mucho más clara para encontrar la solución, mientras que en una búsqueda de una ruta turística se pueden encontrar varias alternativas para encontrar la ruta de destino.

2.5.2 ESPACIO DE ESTADOS

Es uno de los formalismos más comunes para representar problemas. Se va generando conforme avanza la búsqueda, encontrando los nuevos estados que cumplan las condiciones de los diferentes algoritmos de búsqueda hasta llegar al estado objetivo, donde los estados están interconectados por arcos formando una gráfica que capture lo esencial del problema.

La estructura del espacio de estados está compuesta por los siguientes elementos:

Estructura de datos: Representación de árboles y grafos.

Estados: Son representaciones de una situación actual. Contienen toda la información relevante de una localización o ubicación, también son similares a los nodos.

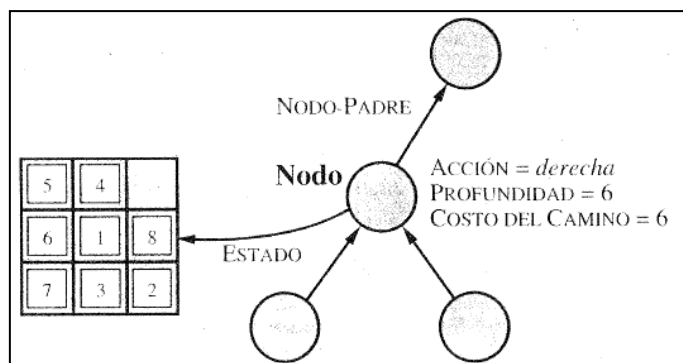


Figura 2.34: Un ejemplo típico del 8-puzzle
Fuente: [Russell Stuart 2004]

En la figura 2.34 también se muestra la diferencia entre un estado y un nodo. Los estados son configuraciones físicas que no incluyen padre, hijo, profundidad y coste de camino, mientras que los nodos poseen tal información.

Operadores: Permiten modificar un estado actual para llevarlo a un nuevo estado mediante un conjunto de acciones. Están definidos en un par de nodos, denominados también como arcos entre nodos.

Árboles y grafos: Están formados por un conjunto de nodos conectados mediante arcos. Los árboles poseen un solo camino que lleva a un nodo, mientras que un grafo posee varios caminos que llevan a un nodo formando un ciclo.

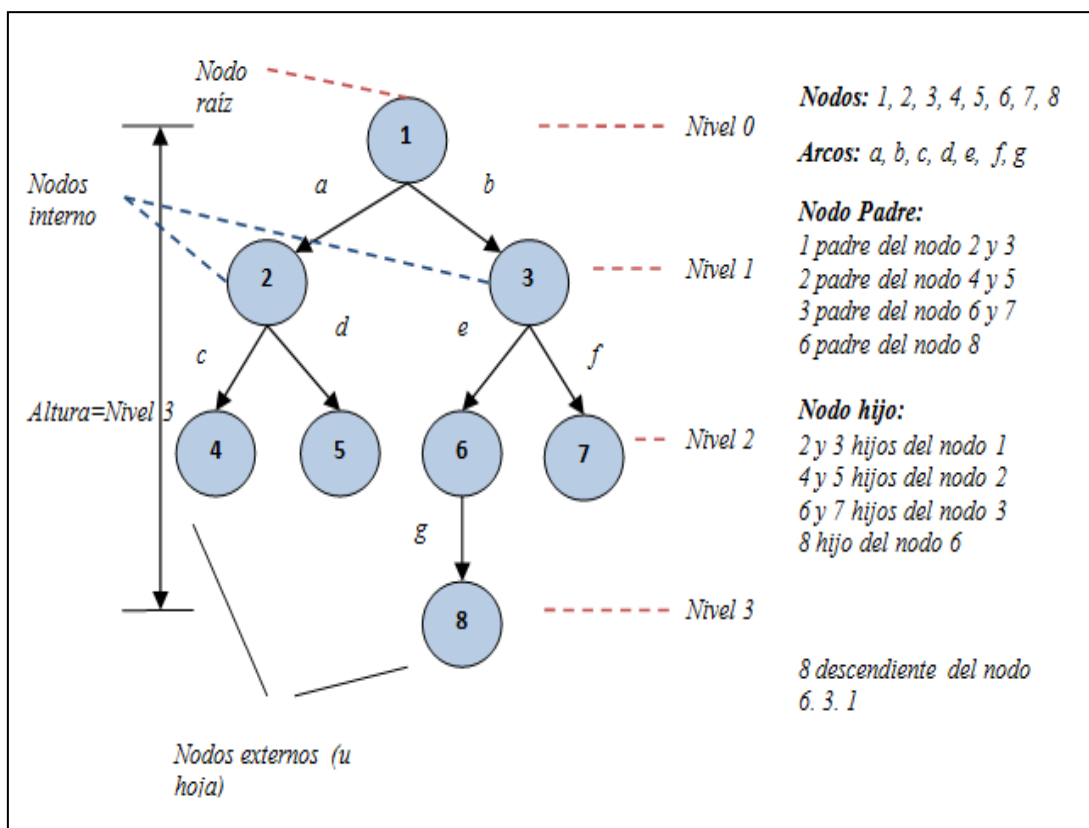


Figura 2.35: Representación de un árbol y sus elementos

Fuente: Autores

2.5.3 ESTRATEGIAS DE BÚSQUEDA

Son secuencias posibles de acciones que permiten conducir desde un estado actual a un estado objetivo, eligiendo y seleccionando el orden de expansión de los nodos de una estructura de datos para la resolución de problemas, tomando en cuenta restricciones de tiempo y de coste computacional.

Existen dos tipos de estrategias de búsquedas para la resolución de problemas, algunas poseen información del problema y otras carecen de ella. Cada estrategia contiene diferentes algoritmos de búsqueda, por esta razón se deben tomar en cuenta consideraciones importantes, tales como: Complejidad temporal, tiempo que tarda en explorar el árbol o grafo de búsqueda, y complejidad espacial, memoria necesaria para almacenar los nodos expandidos.

Antes de definir las estrategias de búsqueda es necesario comprender algunos conceptos fundamentales que ayudarán a un mejor entendimiento en el funcionamiento de los algoritmos de búsqueda que posee cada estrategia.

Estado inicial: Punto de partida para encontrar la solución de un problema.

Estado sucesor: Son los posibles estados alcanzables desde un estado actual.

Test objetivo: Verifica si un estado es el estado objetivo para finalizar la búsqueda del problema.

Coste de camino: Valor numérico que representa el camino entre un estado actual a un nuevo estado encontrado. Este valor se obtiene mediante la sumatoria de las acciones individuales realizadas a lo largo del camino.

Expansión: Se menciona al proceso de generar nodos sucesores desde un nodo padre.

Nodos abiertos: Son los nodos que aún no han sido revisados. Surgen del proceso de expansión y a los cuales se les ha aplicado la función de evaluación $f(n)$.

Nodos cerrados: Son aquellos nodos ya visitados y expandidos por el algoritmo de búsqueda utilizado.

2.5.3.1 ESTRATEGIAS DE BÚSQUEDA NO INFORMADA (CIEGAS)

Son aquellas que no poseen información sobre el problema, tales como: coste de camino o distancia al estado meta para orientar la búsqueda desde un estado inicial al estado final mediante una secuencia de acciones. Estas estrategias tan solo pueden distinguir si un estado es el objetivo o no, realizando una búsqueda exhaustiva en el espacio de búsqueda, siendo también útiles para la resolución de problemas de pequeñas dimensiones. La diferencia de los algoritmos que utilizan esta estrategia es el distinto orden en que se expanden los nodos, entre estas mencionamos dos de las principales.

Primero en anchura

La expansión de los nodos se realiza de una manera exhaustiva de izquierda a derecha empezando en el nodo raíz y luego los nodos generados por éste, sus sucesores y así sucesivamente hasta encontrar la solución. Esta búsqueda no encontrará la solución más óptima pero garantiza encontrar la solución más próxima de existir varias. La figura 2.36 muestra el proceso de búsqueda siguiendo una secuencia de acciones por cada paso realizado.

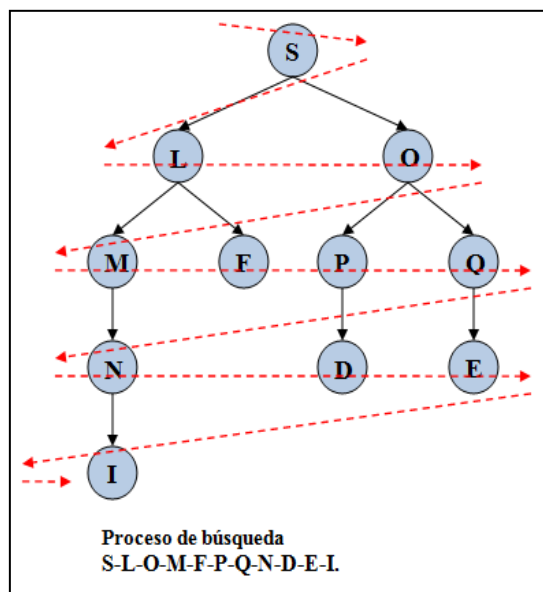


Figura 2.36: Se muestra el proceso de búsqueda para llegar desde S a I utilizando el algoritmo Primero en Anchura.

Fuente: Autores

Primero en profundidad

Realiza una búsqueda exhaustiva descendiendo por cada nivel, generando un único sucesor del nodo en cada paso hasta llegar al nivel más profundo donde se encuentre la solución o un nodo sin sucesores. En caso de encontrar un nodo sin sucesor se retrocede al nodo visitado más cercano para escoger la siguiente rama alternativa del nodo y empezar un nuevo descenso. Esta búsqueda tampoco es la más óptima al igual que la búsqueda primero en anchura, puesto que se generan los mismos nodos en diferente orden.

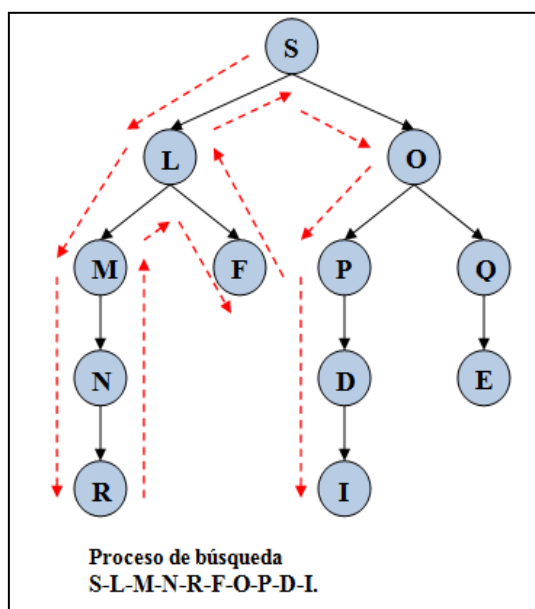


Figura 2.37: La gráfica muestra el descenso realizado por el algoritmo de búsqueda Primero en profundidad de una manera más clara
Fuente: Autores

2.5.3.2 ESTRATEGIAS DE BÚSQUEDA INFORMADA (HEURÍSTICAS)

Son utilizadas para resolver problemas complejos con eficiencia. “Problemas para los que es imposible computacionalmente encontrar la mejor solución para dimensiones elevadas por ese crecimiento exponencial. La única forma de abordarlos es conformarnos con una solución suficientemente buena, y hay varios métodos de búsqueda heurística que proporcionan formas prácticas de hacerlo”¹¹, tratando de optimizar los procesos de búsqueda en tiempo y en espacio utilizando las funciones heurísticas.

Las funciones heurísticas son conjuntos de reglas o métodos que normalmente están representadas por números, utilizados para calcular o evaluar el valor de cada nodo, determinando cuan cerca o lejos se encuentra del nodo objetivo, guiando el proceso de búsqueda en la dirección más prometedora sugiriendo que camino se debe seguir

¹¹ PAJARES MARTINSANZ, Gonzalo y SANTOS PEÑAS, Matilde, *Inteligencia Artificial e Ingeniería del Conocimiento*, 1^{ra}. Edición, Editorial Alfaomega, México, 2006, p.31

cuando hay más de uno. De esta manera se logra reducir el número de nodos expandidos, evaluando las alternativas en función del conocimiento disponible. Una heurística es el conocimiento obtenido ya sea por intuición, pista o experiencia del problema a resolver.

En la figura 2.38 se puede apreciar el coste total de búsqueda con respecto a la cantidad de conocimiento disponible. Cuando la información del problema es nula requiere un elevado coste de control y un excesivo coste de aplicación de reglas¹² por la cantidad de nodos que debe expandir. En caso contrario que se posea una exhaustiva información, se lograría reducir el número de nodos visitados, pero, incrementaría la gestión de la información produciendo un control excesivo y un elevado coste de aplicación de reglas. El objetivo para resolver los problemas de búsquedas es situarse en la parte central de la gráfica (óptima) empleando la información apropiada, disminuyendo de esta manera el proceso de búsqueda para encontrar la solución más óptima.

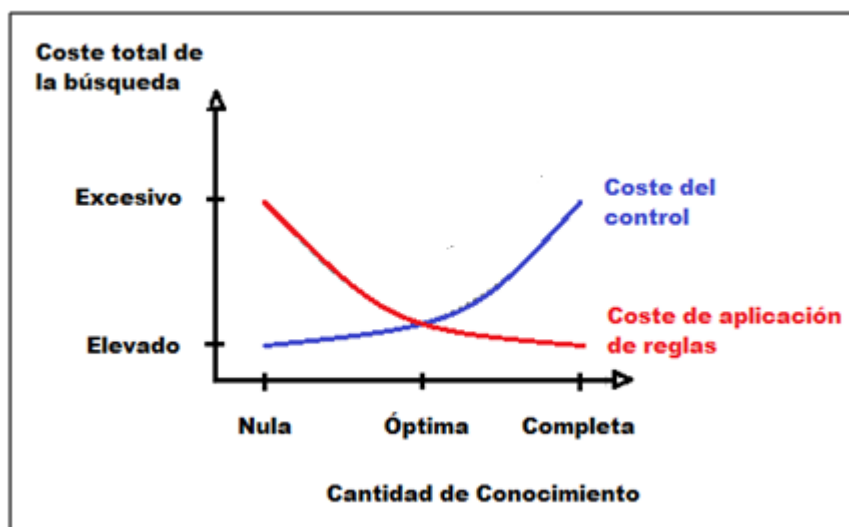


Figura 2.38: Representación del coste total de la búsqueda frente al grado de conocimiento utilizado
Fuente: Autores

¹² Las reglas: son posibles acciones que se puede realizar en un determinado estado para pasar de un punto a otro.

A continuación se mencionan dos de los algoritmos de búsqueda informada más conocidos, utilizados para encontrar la solución más óptima expandiendo la menor cantidad de nodos.

Primero el mejor

Es un método de búsqueda que combina las ventajas de dos algoritmos de búsquedas no informadas en uno solo, siendo estas: la exhaustiva en amplitud y la exhaustiva en profundidad. Hace uso de la información que proviene de la función heurística $h(n)$, para guiar la búsqueda por el camino más prometedor hacia el nodo objetivo.

$h(n)$ Coste en línea recta desde un nodo n a un nodo objetivo, siendo n un nodo cualquiera del espacio de búsqueda.

Su funcionamiento puede resumirse de la siguiente manera.

Algoritmo: Búsqueda primero el mejor

1. Comenzar con ABIERTOS conteniendo sólo el estado inicial.
2. Hasta que se llegue a un objetivo o no queden nodos en ABIERTOS hacer:
 - a) Tomar el mejor nodo de ABIERTOS.
 - b) Generar sus sucesores.
 - c) Para cada sucesor hacer:
 - i. Si no se ha generado con anterioridad, evaluarlo, añadirlo a ABIERTOS y almacenar su padre.
 - ii. Si ya se ha generado antes, cambiar al padre si el nuevo camino es mejor que el anterior. En este caso, se actualiza el coste empleado para alcanzar el nodo y a los sucesores que pudiera tener.¹³

¹³ RICH, Elaine y KNIGHT, Kevin, *Inteligencia Artificial*, 2^{da}. Edición, Editorial McGraw-Hill, Madrid-España, 1994, p.83.

Este algoritmo puede encontrar una solución rápidamente realizando la expansión hacia el nodo más cercano al objetivo. Sin embargo, no toma en cuenta el coste acumulado para llegar a un nodo objetivo desde un nodo inicial, siendo una desventaja que presenta el algoritmo primero el mejor, ya que en ocasiones podría encontrar el camino más largo, sin considerar que pueda existir un camino que lo lleve directamente al nodo meta.

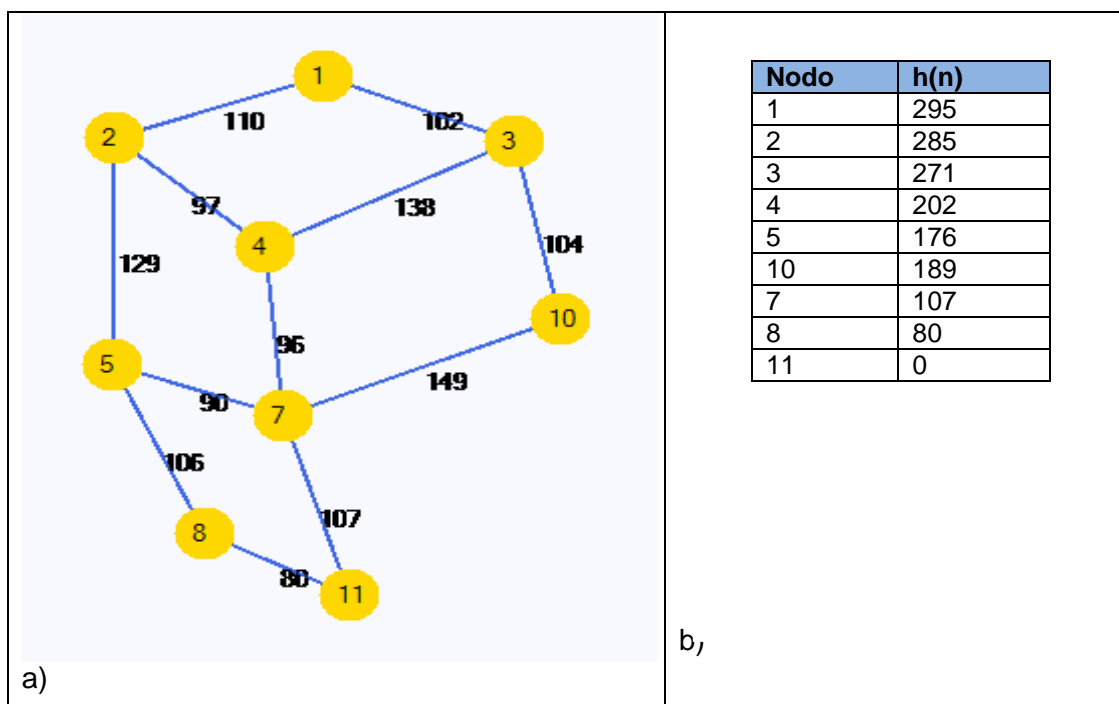
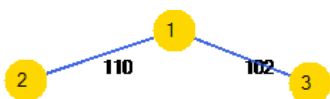


Figura 2.39: a) Grafo, b) Distancia en línea recta al nodo 11

Fuente: Autores

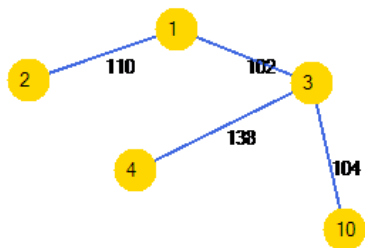
a) Después de expandir 1



285 (2)

271 (3) ←

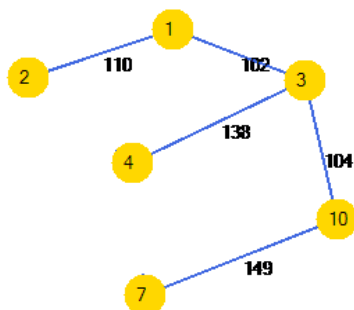
b) Después de expandir 3



202 (4)

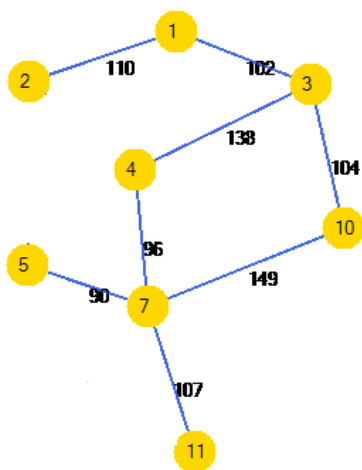
189 (10) ←

c) Después de expandir 10



107 (7) ←

d) Después de expandir 7



176 (5)

202 (4)

0 (11) ←

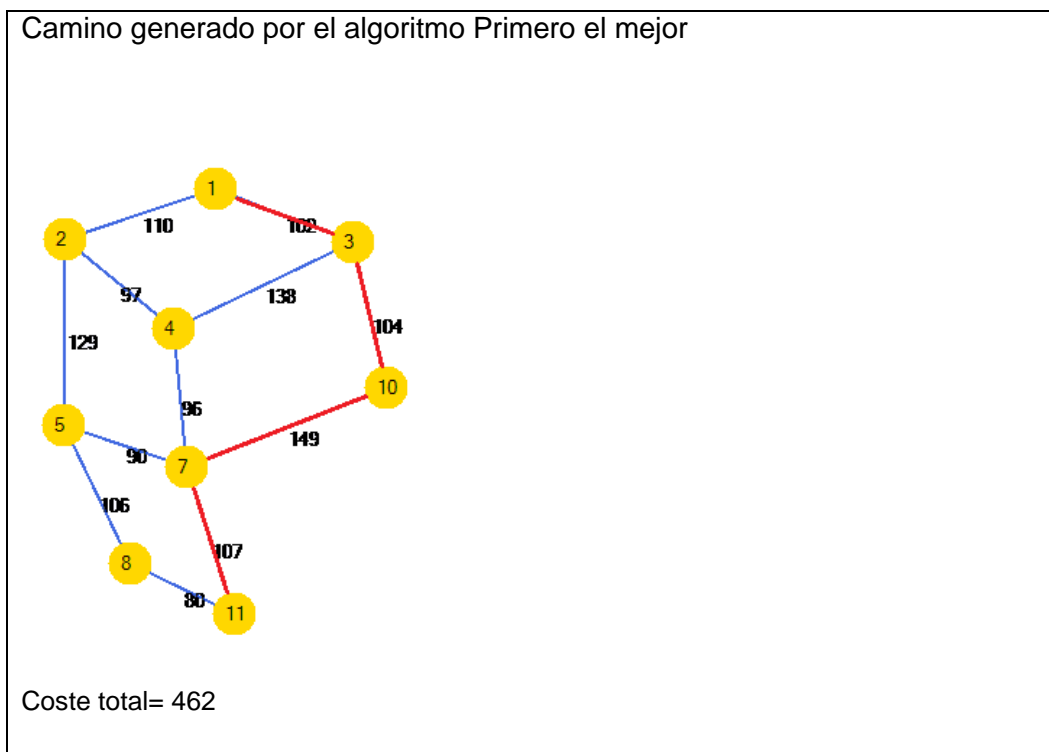


Figura 2.40: Etapas de la búsqueda primero el mejor para ir del nodo 1 al nodo 11
Fuente: Autores

Algoritmo A*

Es uno de más conocidos cuando se requiere encontrar el camino más óptimo desde un punto a otro, reduciendo el número de nodos expandidos. Este algoritmo tiene la propiedad de ser completo (revisar grafos infinitos) y óptimo (tiempo y espacio).

La función de evaluación $f(n)$ para este algoritmo se calcula mediante la suma de dos medidas heurísticas.

$$f(n) = g(n) + h(n)$$

Dónde:

$f(n)$ Coste de camino más barato para llegar al nodo objetivo atravesando los caminos elegidos desde nodo inicio.

$g(n)$ Coste real del camino más óptimo para llegar desde un nodo inicial hasta un nodo n .

$h(n)$ Coste en línea recta real desde un nodo n a un nodo objetivo.

Para cada nodo n se tiene dos funciones: $h'(n)$ estimador de $h(n)$ y $g'(n)$ estimador de $g(n)$ que representan al camino de menor coste para alcanzar el siguiente nodo, por lo tanto una estimación de $f(n)$ será $f'(n)$, donde:

$$f'(n) = g'(n) + h'(n)$$

Los estimadores del nodo n se encuentran en la lista abiertos ordenados de acuerdo al valor mínimo de $f'(n)$.

Admisibilidad

A* es admisible si existe un camino que lo lleve desde el nodo inicio hacia un nodo objetivo encontrando siempre el camino con el menor coste, si se cumple las siguientes condiciones:

1. Cada nodo del grafo debe tener un número finito de sucesores.
2. El coste de cada arco desde un nodo n hasta un nodo n' , sucesor de n , debe ser mayor a una cierta cantidad positiva.
3. La función heurística $h(n)$ debe cumplir la condición $h'(n) \leq h(n)$ para todos los nodos del grafo de búsqueda, siendo $h(n)$ la distancia en línea recta real hacia el objetivo.

Cuando las funciones pueden estar poco informadas, es posible que en ocasiones no siempre se cumpla la condición de admisibilidad, obteniendo lo que se denomina un estimador optimista.

CONSISTENCIA Ó MONOTONÍA

La condición de consistencia o monotonía permite encontrar un camino óptimo para todos los nodos expandidos, si para cada nodo n , cada sucesor de n de n cumple la siguiente condición:

$$h'(n_i) \leq c(n_i, n_j) + h'(n_j)$$

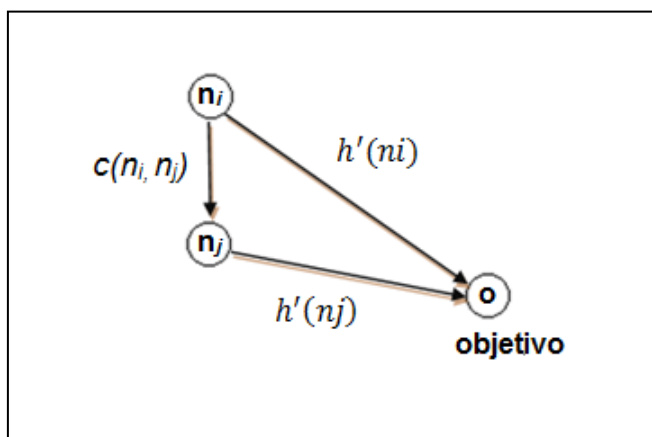


Figura 2.41: Condición de Consistencia
Fuente: [Nilson Nils 2000]

Dónde: $c(n_i, n_j)$ es el coste del arco que va desde nodo n al nodo n' .

El funcionamiento del algoritmo A* puede resumirse de la siguiente manera.

Algoritmo: A*

1. Cree un grafo de búsqueda G , que inicialmente solo contendrá el nodo inicial n_0 . Inserte n_0 en la lista ABIERTOS.
2. Cree la lista CERRADOS, inicialmente vacía.
3. Si la lista ABIERTOS está vacía, el algoritmo termina con fallo, ya que no habremos encontrado ningún nodo objetivo.
4. Extraiga el primer nodo de la lista ABIERTOS e insértelo en la lista CERRADOS a este nodo lo llamaremos n .
5. Si el nodo n es el nodo objetivo el algoritmo termina con éxito, y la solución se obtiene siguiendo el camino desde el nodo n al nodo n_0 , a través de los punteros del grafo G (los punteros describen el árbol de búsqueda y son creados en el paso 7).

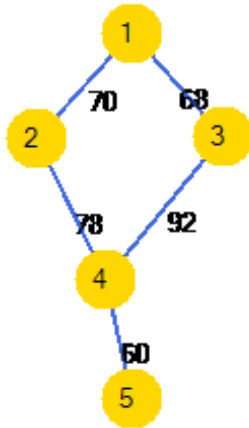
6. *Expanda el nodo n y cree el conjunto M con todos sus sucesores, siempre y cuando estos no sean ascendientes del nodo n en el grafo G . Incorpore los elementos de M como sucesores del nodo n en el grafo G .*
7. *Establezca un puntero a n desde aquellos elementos de M que todavía no hayan sido visitados en el grafo G (es decir, que no estén ni en la lista ABIERTOS ni en la lista CERRADOS). Inserte estos elementos de M en la lista ABIERTOS. Para cada elemento de M que ya estuviera en ABIERTOS o CERRADOS, modifique su puntero para que éste apunte al nodo n , siempre que los mejores caminos que hasta el momento se hayan encontrado hacia dichos nodos pasen por n .
Para cada elemento de M que ya estuviese en CERRADOS, modifique los punteros de sus descendientes en el grafo G , para que apunten hacia atrás a los caminos más eficientes que han sido encontrados hasta el momento a esos descendientes.*
8. *Reordene la lista ABIERTOS según el orden creciente de los valores de la función f' (los casos en los que los valores de f' sean iguales se ordenan según el orden decreciente de la profundidad del nodo).*
9. *Vaya al paso 3.*
[...] la redirección de los punteros de los descendientes de los nodos que ya están en la lista CERRADOS permite economizar los esfuerzos realizados en posteriores pasos del proceso de búsqueda, a costa de aumentar exponencialmente el número de operaciones. Debido a esto, esta parte del paso 7 no se suele implementar.¹⁴

Hay que tomar en cuenta que el paso 7 se aplica al expandir un nodo n , y los punteros son redireccionados siempre y cuando se encuentren aquellos nodos que ya estén en ABIERTOS o en CERRADOS con el camino más óptimo.

La figura 2.42 muestra el redireccionamiento realizado en un nodo abierto, donde existen dos caminos que llegan al mismo nodo, un camino anteriormente analizado con un coste de 160 y el nuevo camino encontrado con un coste de 148. Si el coste del camino encontrado es menor que el analizado anteriormente se redirecciona el

¹⁴ NILSON, Nils J., *Inteligencia Artificial Una Nueva Síntesis*, 1^{ra}. Edición, Editorial McGraw-Hill, Madrid-España, 2000, p.130, 131.

puntero al nodo n y se actualiza su función de coste mínimo a 208 en la lista ABIERTOS, caso contrario se abandona este paso.



Distancias en línea recta al nodo 5

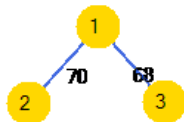
Nodo	h(n)
1	181
2	137
3	138
4	60
5	0

a) Estado inicial



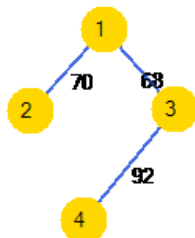
Abiertos	Cerrados	Grafo de búsqueda G
$f(n) = g(n) + h(n)$ $181 = 0 + 181$ (1)		

b) Después de expandir 1



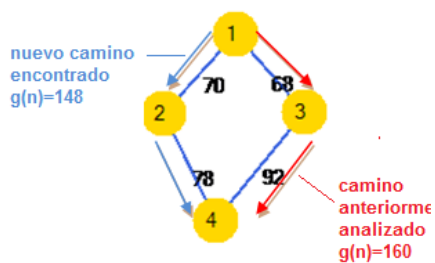
Abiertos	Cerrados	Grafo de búsqueda G
$206 = 68 + 138$ (3)	181 (1)	$3 \rightarrow 1$
$207 = 70 + 137$ (2)		$2 \rightarrow 1$

c) Después de expandir 3



Abiertos	Cerrados	Grafo de búsqueda G
$207 = 70 + 137$ (2)	181 (1)	$3 \rightarrow 1$
$220 = 160 + 60$ (4)	206 (3)	$2 \rightarrow 1$
		$4 \rightarrow 3$

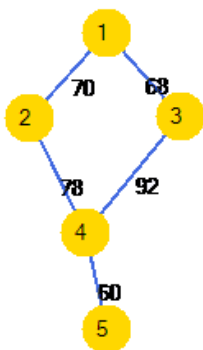
d) Después de expandir 2



Abiertos	Cerrados	Grafo de búsqueda G
$208=148+60$ (4)	181 (1)	$3 \rightarrow 1$
	206 (3)	$2 \rightarrow 1$
	207 (2)	$4 \rightarrow 3$ X $4 \rightarrow 2$

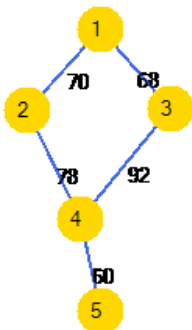
Nuevo camino ($g(n)=148$) < camino anterior ($g(n)=160$), entonces redireccionar el puntero al nuevo camino y actualizar la función de $f(n)=220$ a $f(n)=208$

e) Después de expandir 4



Abiertos	Cerrados	Grafo de búsqueda G
$208=208+0$ (5)	181 (1)	$3 \rightarrow 1$
	206 (3)	$2 \rightarrow 1$
	207 (2)	$4 \rightarrow 2$
	208 (4)	$5 \rightarrow 4$

f) Después de expandir 5



Abiertos	Cerrados	Grafo de búsqueda G
	181 (1)	$3 \rightarrow 1$
	206 (3)	$2 \rightarrow 1$
	207 (2)	$4 \rightarrow 2$
	208 (4)	$5 \rightarrow 4$
	208 (5)	

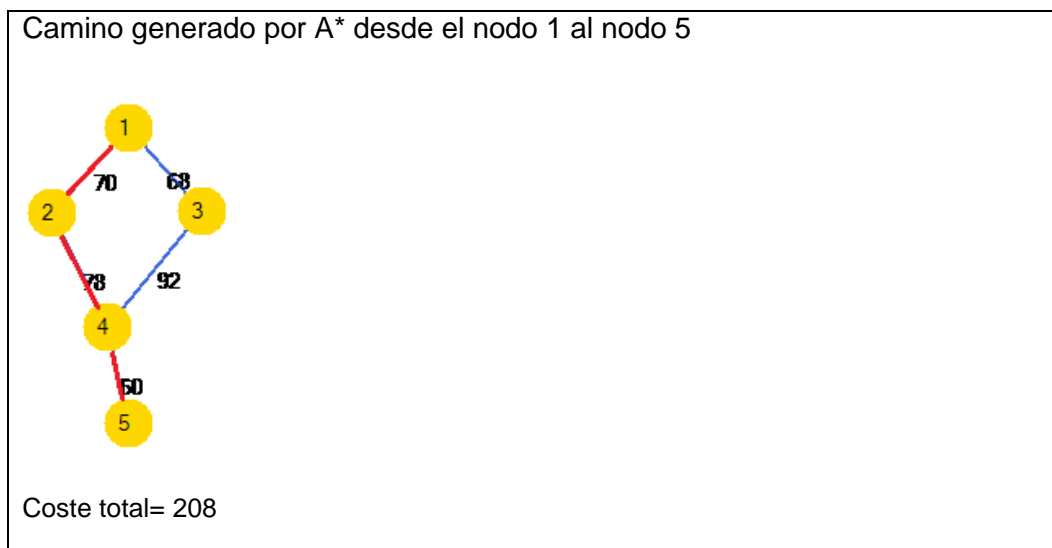



Figura 2.42: Redireccionamiento realizado en un nodo que se encuentra en ABIERTOS
Fuente: Autores

El redireccionamiento realizado en un nodo cerrado, se realiza de la misma manera que al encontrar un nodo abierto, con la diferencia que este nodo encontrado es extraído de la lista CERRADOS e insertada en lista ABIERTOS actualizando su función de coste mínima.

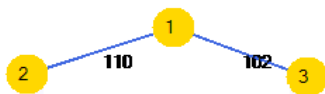
En cuanto a los sucesores de los nodos abiertos o cerrados encontrados, no hay la necesidad de actualizar sus funciones de coste mínimo, ya que se irán actualizando durante el proceso de búsqueda.

Ejemplo1:

Los datos para el siguiente ejemplo han sido tomados de la figura 2.39.

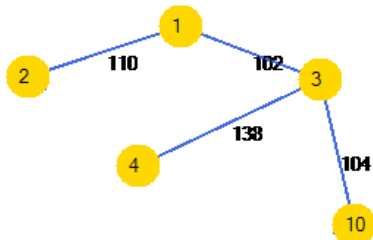
a) Estado inicial			
			
Abiertos	Cerrados	Grafo de búsqueda G	
$f(n) = g(n) + h(n)$ $295 = 0 + 295 (1)$			

b) Después de expandir 1



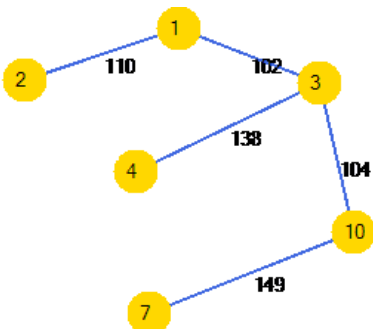
Abiertos	Cerrados	Grafo de búsqueda G
$373=102+271$ (3)	295 (1)	$2 \rightarrow 1$
$395=110+285$ (2)		$3 \rightarrow 1$

c) Después de expandir 3



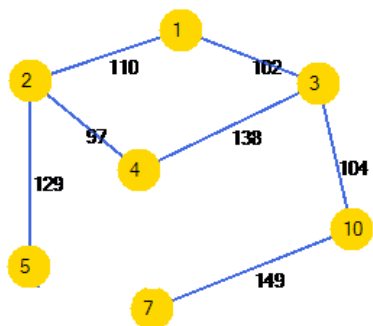
Abiertos	Cerrados	Grafo de búsqueda G
$395=206+189$ (10)	295 (1)	$2 \rightarrow 1$
$395=110+285$ (2)	373 (3)	$3 \rightarrow 1$
$442=240+202$ (4)		$4 \rightarrow 3$
		$10 \rightarrow 3$

d) Después de expandir 10



Abiertos	Cerrados	Grafo de búsqueda G
$395=110+285$ (2)	295 (1)	$2 \rightarrow 1$
$442=240+202$ (4)	373 (3)	$3 \rightarrow 1$
$462=355+107$ (7)	395 (10)	$4 \rightarrow 3$
		$10 \rightarrow 3$
		$7 \rightarrow 10$

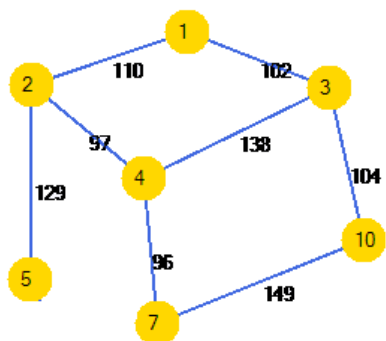
e) Después de expandir 2



Abiertos	Cerrados	Grafo de búsqueda G
$409=207+202$ (4)	295 (1)	$2 \rightarrow 1$
$415=239+176$ (5)	373 (3)	$3 \rightarrow 1$
$462=355+107$ (7)	395 (10)	$4 \rightarrow 3$ X $4 \rightarrow 2$
	395 (2)	$10 \rightarrow 3$
		$7 \rightarrow 10$
		$5 \rightarrow 2$

Redireccionar el puntero del nodo 4 al nodo 2 puesto que el coste acumulado es menor al pasar por los nodos (1-2-4=207) mientras el coste acumulado al pasar por nodos (1-3-4=240) es mayor, además cambiar el valor de la función de 442 (4) a 409 (4).

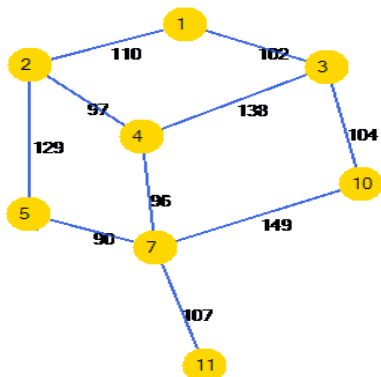
f) Después de expandir 4



Abiertos	Cerrados	Grafo de búsqueda G
410= 303+107 (7)	295 (1)	2 → 1
415= 239+176 (5)	373 (3)	3 → 1
	395 (10)	4 → 2
	395 (2)	10 → 3
	409 (4)	7 → 10 X 7 → 4
		5 → 2

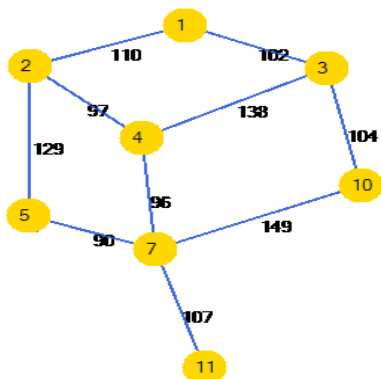
Redireccionar el puntero del nodo 7 al nodo 4 puesto que el coste acumulado es menor al pasar por los nodos (1-2-4-7=303) mientras el coste acumulado al pasar por nodos (1-3-10-7=355) es mayor, además cambiar el valor de la función de 462 (7) a 410 (7).

g) Después de expandir 7



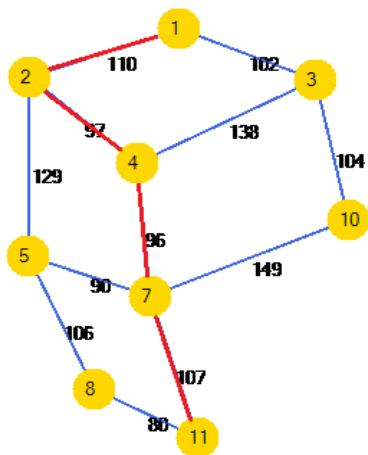
Abiertos	Cerrados	Grafo de búsqueda G
410=410+ 0(11)	295 (1)	2 → 1
415= 239+176 (5)	373 (3)	3 → 1
	395 (10)	4 → 2
	395 (2)	10 → 3
	409 (4)	7 → 4
	410 (7)	5 → 2
		11 → 7

h) Después de expandir 11



Abiertos	Cerrados	Grafo de búsqueda G
415= 239+176 (5)	295 (1)	2 → 1
	373 (3)	3 → 1
	395 (10)	4 → 2
	395 (2)	10 → 3
	409 (4)	7 → 4
	410 (7)	5 → 2
	410 (11)	11 → 7

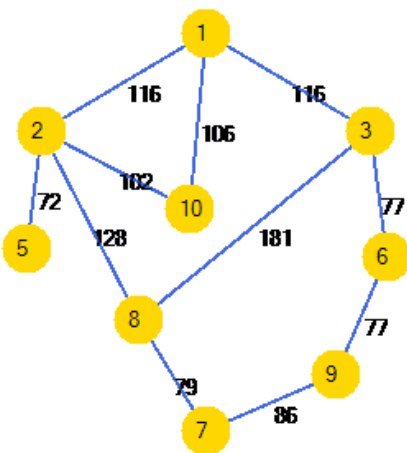
Camino generado por A* desde el nodo 1 al nodo 11



Coste total = 410

Figura 2.43: Etapas de la búsqueda A* desde el nodo 1 al nodo 11
Fuente: Autores

Ejemplo 2:



Distancias en línea recta al nodo 7

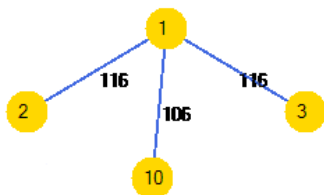
Nodo	h(n)
1	240
2	207
3	207
5	155
6	152
7	0
8	79
9	86
10	134

a) Estado inicial



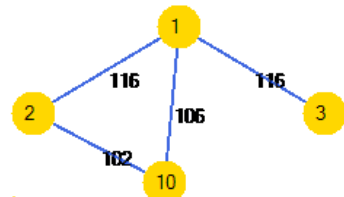
Abiertos	Cerrados	Grafo de búsqueda G
$f(n) = g(n) + h(n)$ $240 = 0 + 240$ (1)		

b) Después de expandir 1



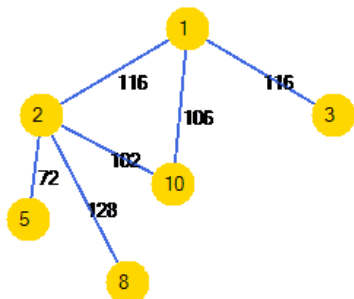
Abiertos	Cerrados	Grafo de búsqueda G
$240 = 106 + 134(10)$	240 (1)	$2 \rightarrow 1$
$323 = 116 + 207(3)$		$3 \rightarrow 1$
$323 = 116 + 207(2)$		$10 \rightarrow 1$

c) Después de expandir 10



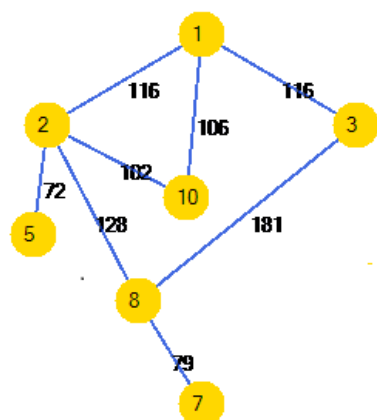
Abiertos	Cerrados	Grafo de búsqueda G
$323 = 116 + 207(2)$	240 (1)	$2 \rightarrow 1$
$323 = 116 + 207(3)$	240 (10)	$3 \rightarrow 1$
		$10 \rightarrow 1$

d) Después de expandir 2



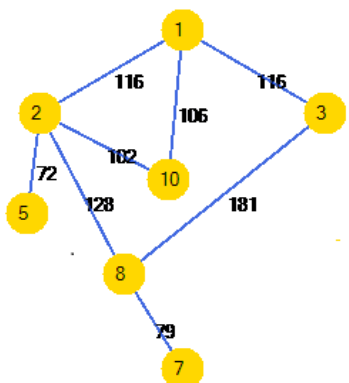
Abiertos	Cerrados	Grafo de búsqueda G
$323 = 244 + 79(8)$	240 (1)	$2 \rightarrow 1$
$323 = 116 + 207(3)$	240 (10)	$3 \rightarrow 1$
$343 = 188 + 155(5)$	323 (2)	$10 \rightarrow 1$
		$5 \rightarrow 2$
		$8 \rightarrow 2$

e) Después de expandir 8



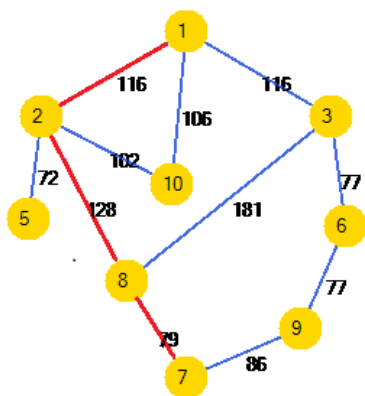
Abiertos	Cerrados	Grafo de búsqueda G
$323 = 323 + 0(7)$	240 (1)	$2 \rightarrow 1$
$323 = 116 + 207(3)$	240 (10)	$3 \rightarrow 1$
$343 = 188 + 155(5)$	323 (2)	$10 \rightarrow 1$
	323 (8)	$5 \rightarrow 2$
		$8 \rightarrow 2$
		$7 \rightarrow 8$

f) Después de expandir 7



Abiertos	Cerrados	Grafo de búsqueda G
$323 = 116 + 207(3)$	240 (1)	$2 \rightarrow 1$
$343 = 188 + 155(5)$	240 (10)	$3 \rightarrow 1$
	323 (2)	$10 \rightarrow 1$
	323 (8)	$5 \rightarrow 2$
	323 (7)	$8 \rightarrow 2$
		$7 \rightarrow 8$

Camino generado por A* desde el nodo 1 al nodo 7



Coste total = 323

Figura 2.44: Etapas de la búsqueda A* desde el nodo 1 al nodo 11
Fuente: Autores

La única diferencia entre A* y el algoritmo primero el mejor, es que éste garantiza encontrar la solución más óptima, siempre que esta exista, debido a una evaluación más eficiente que realiza en los costes.

CAPÍTULO 3

ANÁLISIS Y DESARROLLO

3.1 ANÁLISIS

3.1.1 ROBOTINO

Es un robot móvil omnidireccional desarrollado y fabricado por la empresa FESTO para fines de formación profesional. Su hardware está compuesto por una amplia gama de tecnología, tales como: Actuadores eléctricos, sensores, tecnologías de regulación, técnicas de procesamiento de imágenes y programación, lo que le permite al robot móvil Robotino ser autónomo.



Figura 3.1: Robotino de FESTO
Fuente: Manual Robotino, Festo, 2007

Este robot móvil es utilizado con fines didácticos para la automatización y optimización industrial aplicando conocimiento tanto de:

- **Mecánica:** Montar y desmontar los componentes del Robotino, con el fin de desarrollar la creatividad de los estudiantes en la construcción de robots móviles.
- **Electricidad:** Aprender las conexiones de los componentes eléctricos que posee Robotino.
- **Sensores:** Controlar el seguimiento de trayectorias evitando las posibles colisiones.
- **Sistemas de control por realimentación:** Controlar los accionamientos omnidireccionales para que el Robotino se desplace en cualquier orientación.
- **Utilización de interfaces de comunicación:** Establecer una comunicación desde la PC al Robotino utilizando WLAN.
- **Montaje e integración de nuevos sensores y dispositivos manipuladores:** Comprender la integración de componentes adicionales que posee el Robotino.
- **Programación de algoritmos:** Crear un sistema de navegación autónomo utilizando una librería desarrollada bajo C++.

El robot móvil Robotino cuenta con una estructura modular que facilita al estudiante un aprendizaje interactivo, observando el comportamiento de cada uno de sus componentes a través de una interfaz de usuario. Además cuenta con su propio simulador de experimentación virtual en 3D que le permite al estudiante realizar algunas pruebas de su funcionamiento sin la necesidad de utilizar el Robotino directamente.

La figura 3.2 muestra un entorno virtual en 3D donde el estudiante puede realizar simulaciones como: seguimiento de un camino marcado utilizando los sensores ópticos o mediante una cámara web empleando procesamiento de imágenes, también permiten al estudiante realizar simulaciones de detección y evasión de obstáculos empleando los sensores infrarrojos de distancia, los cuales proporcionan datos que son empleados para calcular la distancia a la que se encuentra el Robotino de un objeto.

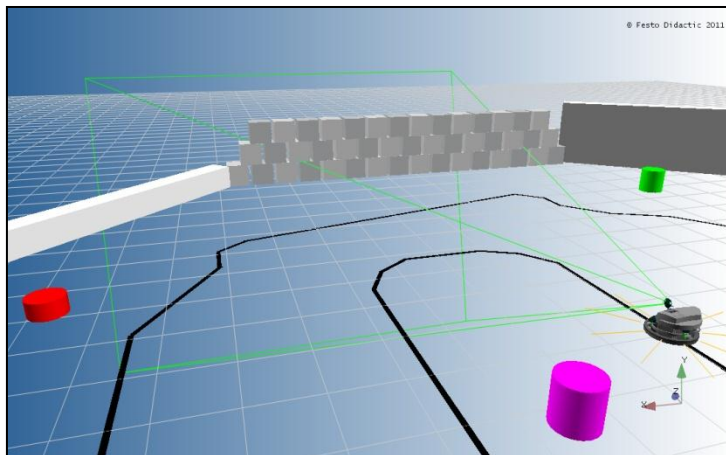


Figura 3.2: Entorno 3D del Robotino (Robotino Sim)
Fuente: Autores

3.1.1.1 CHASIS Y BATERÍAS

El chasis está formado por una plataforma de acero inoxidable soldada con láser. Esta también cuenta con dos asas en sus extremos para agarre, siendo de gran utilidad para evitar el daño del puente de mando y de los sensores al momento de trasladar al Robotino de un lugar a otro. Está diseñado de tal forma que cada componente calce en un determinado espacio, en el caso que se desee acoplar sensores o actuadores, también cuenta con un espacio adicional para aquellos.

Las baterías del Robotino son recargables a 12 V DC con una capacidad de 4 Ah cada una. Estas baterías suministran energía ininterrumpida por alrededor de dos horas antes de ser recargadas.

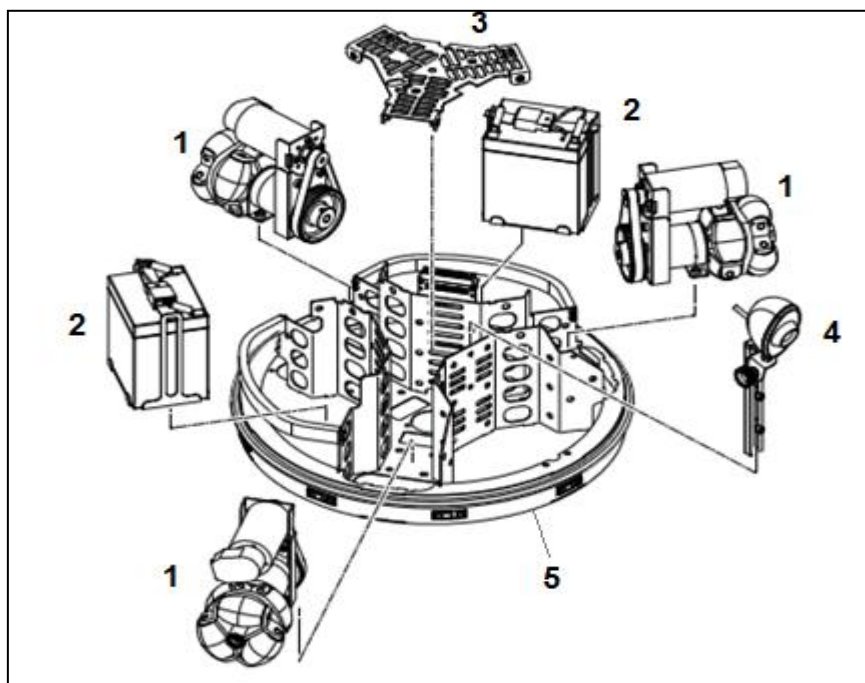


Figura 3.3: Chasis y componentes del Robotino
Fuente: Manual Robotino, Festo, 2007

No.	Componentes
1	motores DC
2	Baterías
3	Placa de fondo con liston protector
4	Plataforma de trabajo con webcam
5	Chasis

3.1.1.2 UNIDAD DE ACCIONAMIENTO

El Robotino cuenta con 3 unidades de accionamiento omnidireccional ubicados a 120° entre sí bajo el chasis, que le permiten desplazarse en cualquier orientación, cada unidad de accionamiento está compuesta por los siguientes componentes:

- **Motor DC:** Cada motor se mueve a una tensión nominal de 24 VDC y una velocidad nominal de 3600 RPM¹⁵, además su velocidad puede ser regulada por un regulador PID¹⁶ a través de la placa de circuito de E/S.

¹⁵ RPM-Revoluciones por minuto.

¹⁶ PID-Proporcional, Integral, Derivativo.

- **Rodillos omnidireccionales:** El rodillo tiene un diámetro de 80 mm para soportar una carga máxima de 40 kg.
- **Correa dentada:** Permite la transmisión de la energía mecánica entre el motor y el reductor.
- **Encoder incremental:** Es empleado para transformar el movimiento angular en pulsos digitales.
- **Reductor con una relación de reducción de 16:1**¹⁷

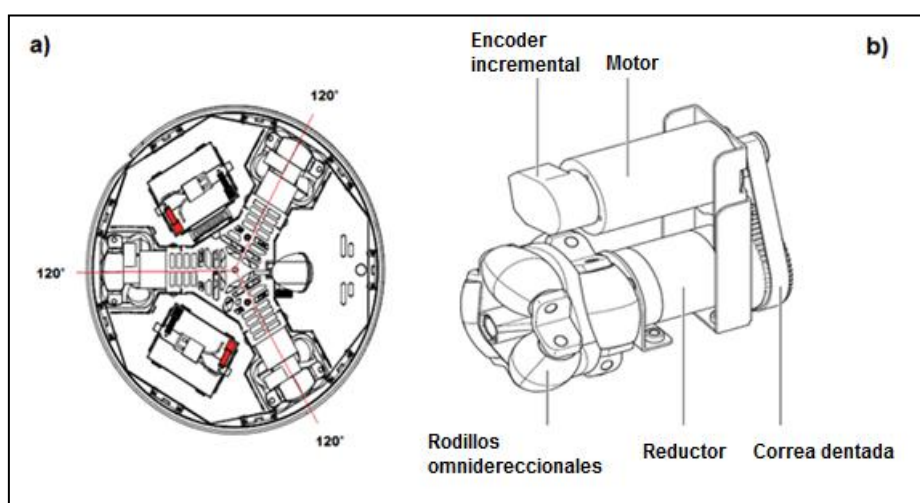


Figura 3.4: a) Ubicación de las unidades de accionamiento, b) Unidad de accionamiento y sus partes

Fuente: Manual Robotino, Festo, 2007

3.1.1.3 SENSORES

Sensores de medición de distancia por infrarrojos

EL Robotino cuenta con 9 sensores infrarrojos para medir distancias desde 4cm a 40cm, los cuales están montados al chasis del robot formando un ángulo de 40° del uno al otro. Esto permite al robot revisar un área más extensa al momento de desplazarse y evadir obstáculos. Cada estado del sensor puede ser consultado individualmente cuando sea necesario.

¹⁷ Reducción de 16:1, ver anexo 3

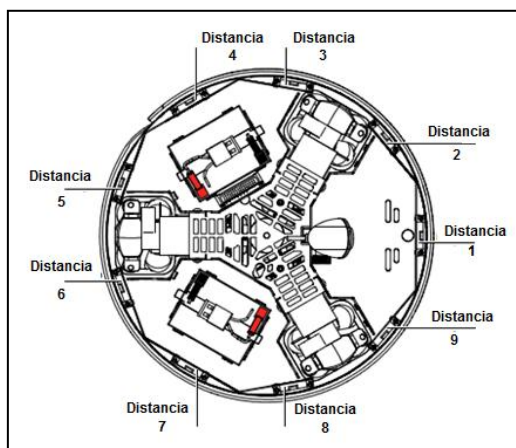


Figura 3.5: Distribución de los sensores de distancia
Fuente: Manual Robotino, Festo, 2007

Estos sensores envían una señal analógica de tensión (0 – 2.5V) que varía de acuerdo a la distancia en la que se encuentre el obstáculo. Estos valores son útiles cuando se requiere calcular distancias precisas.

Encoder incremental

Nos indica el ángulo girado por el eje del motor, con el fin de conocer con precisión el posicionamiento del Robotino de FESTO. Esta compuesto por un diodo emisor de luz, un disco de metal con ranuras y una matriz fotoreceptor.

El encoder incremental proporciona normalmente dos formas de ondas cuadradas (canal A y canal B) desfazadas entre sí a 90°. Un canal A que permite conocer la velocidad de rotación del motor y un canal B que discrimina el sentido del giro en base a las señales producidas en ambos canales. Además cuenta con un canal I que indica el giro completo del eje del motor.

Su funcionamiento se basa en la rotación de un disco de metal con ranuras transparentes iluminadas mediante un emisor de luz. Estas ranuras proyectadas por la luz son receptadas por la matriz fotoreceptora que codifica las variaciones de luz mediante un circuito eléctrico a señales digitales.

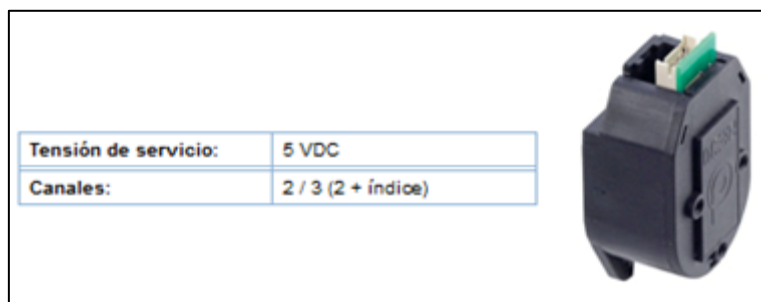


Figura 3.6: Encoder incremental RE30
Fuente: Manual Robotino, Festo, 2007

Sensor anticolidión

Formado por una banda de detección fijada alrededor del aro que rodea el chasis, en el interior de esta banda se encuentran dos superficies conductoras que mantienen una determinada distancia entre sí. En el momento que estas superficies conductoras entran en contacto por una mínima presión ejercida, se envía una señal binaria a la unidad de control para detener completamente al Robotino.

Sensor de proximidad inductivo analógico M12

Sensor con blindaje metálico para la detección de franjas metálicas en el piso a una distancia de 0 a 6mm. Envía una señal analógica de 0 a 10V que varía de acuerdo a la altura que se encuentra del piso y al tipo de material que se vaya utilizar.



Figura 3.7: Sensor inductivo
Fuente: [Festo Didactic]

El sensor contiene cuatro cables para su funcionamiento, dos para alimentación, Vcc (15 – 30V) y Gnd, y dos para la transmisión de señales, tensión (0 - 10V) e intensidad (4 – 20mA). En la figura 3.8 se muestra la conexión del sensor inductivo a la regleta de conexiones de componentes adicionales que posee el Robotino.

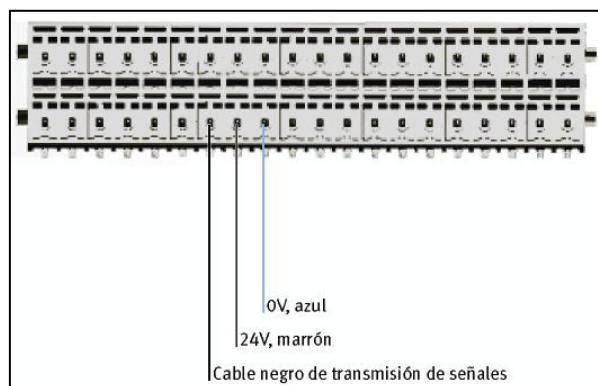


Figura 3.8: Conexión del sensor inductivo a la interface E/S
Fuente: Manual Robotino, Festo, 2007

Sensores de reflexión directa SOEG-L-Q30-P-A-S-2L

Son sensores que inciden luz difusa¹⁸. Cuentan con un emisor de luz roja visible y un receptor que detecta la luz reflejada utilizando un cable de fibra óptica para cada uno, evitando de esta manera pérdidas de luz, con el fin de obtener una mayor calidad en la detección de marcas en el piso utilizando Robotino de FESTO. La detección varía de acuerdo al color del material y al tipo de superficie en la que se trabaje. Sin embargo, estos sensores no pueden detectar diferencias graduales en la luz reflejada, es decir el sensor envía una señal binaria de 0 o 1 dependiendo de la cantidad de luz reflejada.

¹⁸ Se denomina luz difusa a la luz que incide sobre los objetos desde múltiples ángulos, proporcionando una iluminación más homogénea y haciendo que las sombras sean menos nítidas cuanto más lejos esté un objeto de la superficie que oscurece.



Figura 3.9: Sensor óptico
Fuente: [Festo Didactic]

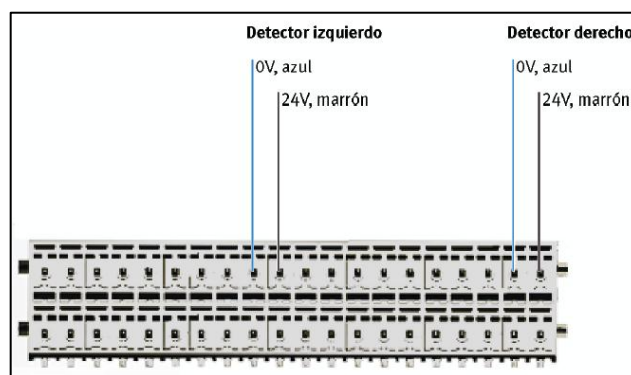


Figura 3.10: Conexión eléctrica del sensor óptico
Fuente: Manual Robotino, Festo, 2007

Estos sensores poseen un potenciómetro que puede ser utilizado para reducir o incrementar la distancia máxima de reflexión, soportan una tensión de 10 a 30 VDC y proporcionan salidas tipo NPN (NC/NA) o PNP (NA/NC). Su tiempo de respuesta es de 0,5ms ante las variaciones de luz ocasionadas por el tipo de material detectado.

La figura 3.11 muestra la curva de aproximación que genera el sensor óptico cuando un objeto suficientemente reflectante penetra en la zona de detección.

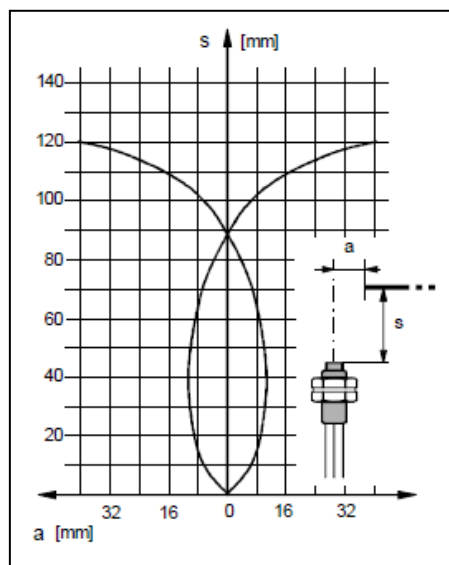


Figura 3.11: Curva de aproximación
Fuente: [Festo Didactic]

3.1.1.4 PUENTE DE MANDO

Contiene los componentes más sensibles del Robotino, tales como la unidad de control, teclado de membrana y display, tarjeta compact flash y la interfaz E/S. Estos a su vez se conectan con los demás componentes mediante un conector.

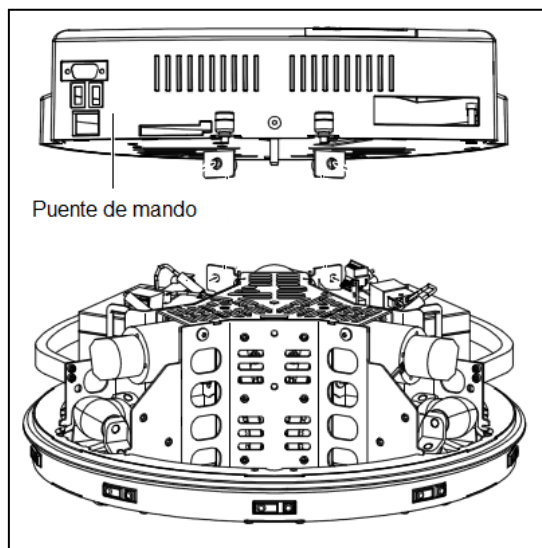


Figura 3.12: Puente de mando del Robotino
Fuente: Manual Robotino, Festo, 2007

3.1.1.5 UNIDAD DE CONTROL

La unidad de control permite la entrada y salida de datos provenientes de dispositivos de entrada, una vez decodificados estos datos son ejecutados para realizar la tarea asignada.

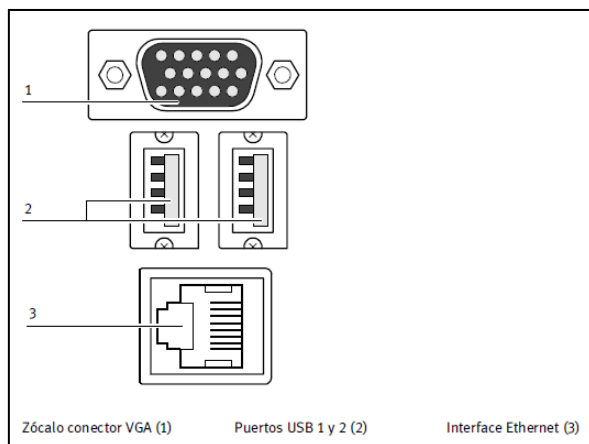


Figura 3.13: Interfaz de E/S
Fuente: Manual Robotino, Festo, 2007

La figura 3.13 muestra la unidad de control del Robotino que está equipada con las siguientes interfaces de entrada y salida de datos.

- **Ethernet:** Permite al usuario acceder al sistema del Robotino desde una computadora.
- **USB:** Esta interfaz permite conectar componentes, tales como una cámara, mouse y un teclado para el ingreso de datos o manipulación del Robotino.
- **VGA:** Permite conectar una pantalla en el Robotino para observar lo que se está programando sin la necesidad de un ordenador.

3.1.1.6 TECLADO DE MEMBRANA Y DISPLAY

Se encuentra ubicada sobre la parte superior del Robotino. El teclado de membrana permite seleccionar algunas opciones, tales como: Programas de demostraciones

integradas en el Robotino (Seguir una línea marcada, evitar obstáculos, etc.), configurar su dirección IP siendo esta dinámica (DHCP) o estática y visualizar el estado de las baterías mediante un display como se muestra en la figura 3.14.

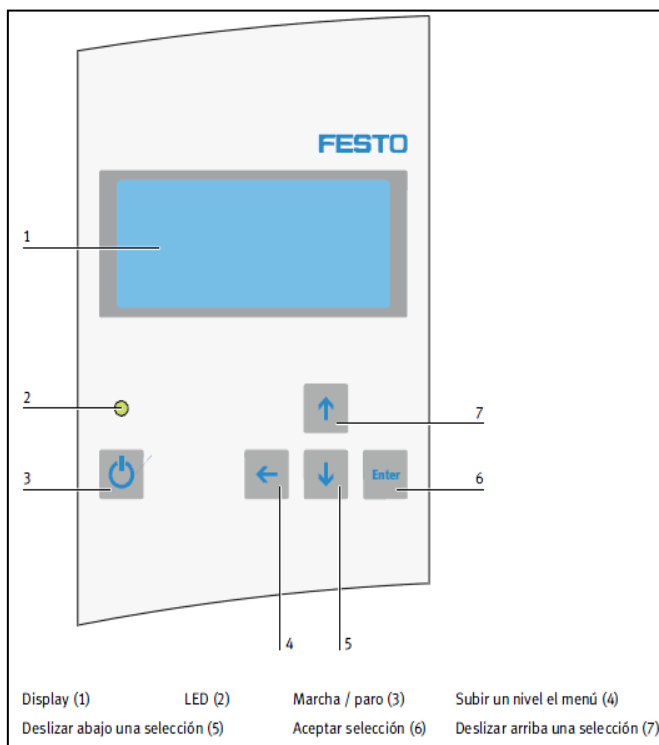


Figura 3.14: Teclado de membrana y display

Fuente: Manual Robotino, Festo, 2007

3.1.1.7 TARJETA COMPACT FLASH

Es uno de los dispositivos más importantes que contiene el Robotino. Esta tarjeta posee un sistema operativo bajo GNU/Linux, API para C++ y programas de demostración del Robotino.

Todas las demostraciones desarrolladas que contiene esta tarjeta son ejecutadas bajo C++ desde el propio Robotino. Esto permite realizar programas personalizados sin la necesidad de un ordenador externo.



Figura 3.15: Tarjeta Compact Flash
Fuente: [Festo Didactic]

3.1.1.8 INTERFACE E/S

Esta interfaz permite la conexión de sensores y actuadores adicionales en el Robotino, tales como: Sensores ópticos, sensores inductivos, relés, etc. En la figura 3.16 se puede apreciar lo siguiente:

- 8 entradas analógicas de 0 a 10 volts (AIN0 hasta AIN7)
- 8 entradas digitales (DI0 hasta DI7)
- 8 salidas digitales (DO0 hasta DO7)
- 2 relés para actuadores adicionales NA o NC (REL0 y REL1)

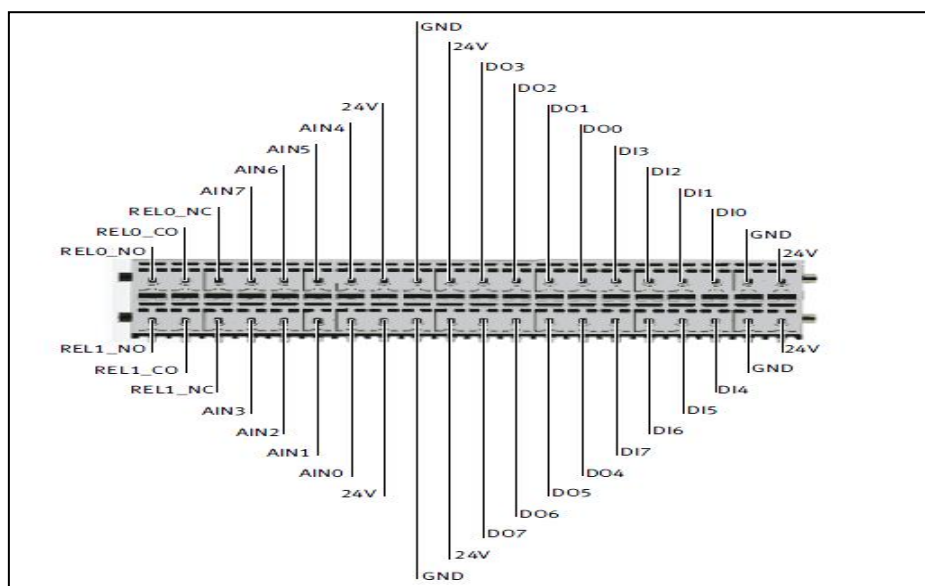


Figura 3.16: Asignación de bornes del Interfaz de E/S
Fuente: Manual Robotino, Festo, 2007

3.1.1.9 MÓDULO TARJETA CIRCUITO DE ENTRADA

Permite la comunicación y el control de todos los módulos incluidos en el Robotino. Toda la información recolectada por medio de sus sensores es enviada a esta tarjeta, que hace uso del sistema para controlar todos los componentes en tiempo real.



Figura 3.17: Tarjeta circuito de entrada
Fuente: [Festo Didactic]

3.1.1.10 PUNTO DE ACCESO LAN INALÁMBRICO

Permite al usuario acceder directamente al control del Robotino mediante una conexión de red inalámbrica (WLAN). Esta conexión permite la comunicación desde el robot a aplicaciones Web o entornos de programación utilizando el API para el Robotino por medio de una dirección IP, ofreciendo una movilidad sin la necesidad de cables para el envío y recepción de la información, llegando a distancias de hasta 100m.

El módulo Air Live incorporado en el Robotino permite conectarse ya sea en modo AP (punto de acceso), modo cliente y modo RT (Router).

En la figura 3.18 se muestra al Robotino en la modalidad WAP (Wireless Access Point), ya que dispone de su propio servidor WLAN. Esto permite establecer la conexión con cualquier PC que disponga de comunicación WLAN

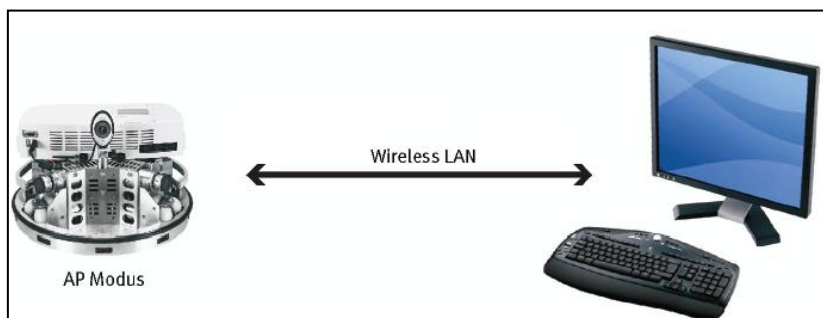


Figura 3.18: Comunicación a través de Wlan
Fuente: Manual Robotino, Festo, 2007

3.1.1.11 LABTEC WEBCAM PRO

La webcam permite observar una parte del entorno en tiempo real y ver lo que sucede con gran precisión. Puede ser utilizada para buscar objetos y colores específicos, como también para realizar tareas de seguimiento de caminos, etc.

La imagen capturada por la webcam puede ser transmitida a la PC desde el Robotino a través de la WLAN en formato JPEG, con el fin de ser procesada por aplicaciones externas como Java, Matlab, Labview, c#, C++, etc., y de esta manera extraer las características relevantes útiles para el cumplimiento de las tareas encomendadas. Generalmente la webcam está fijada al frente del Robotino conectado mediante una interfaz USB a la unidad de control del robot para su funcionamiento.

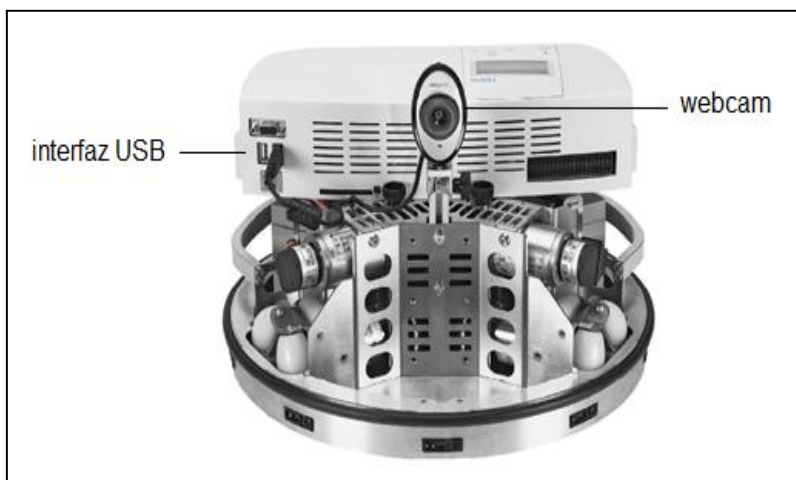


Figura 3.19: Webcam del Robotino
Fuente: Manual Robotino, Festo, 2007

3.1.2 API (INTERFAZ DE PROGRAMACIÓN DE APLICACIONES)

El API es una interfaz de comunicación entre el software y los componentes del Robotino, permitiendo el pleno acceso a los sensores y actuadores. Existe un API para Windows y otro para GNU/Linux, cada API posee varias librerías que el programador puede utilizar para desarrollar aplicaciones en diferentes lenguajes de programación, tales como .Net, C++, C, C# y Java, haciendo el control del Robotino más personalizado.

En esta tesis se ha utilizado el API para Windows. El API contiene la librería `rec.robotino.com` desarrollada para aplicaciones bajo C#. NET 2010, que a su vez contiene las clases para el control de cada componente.

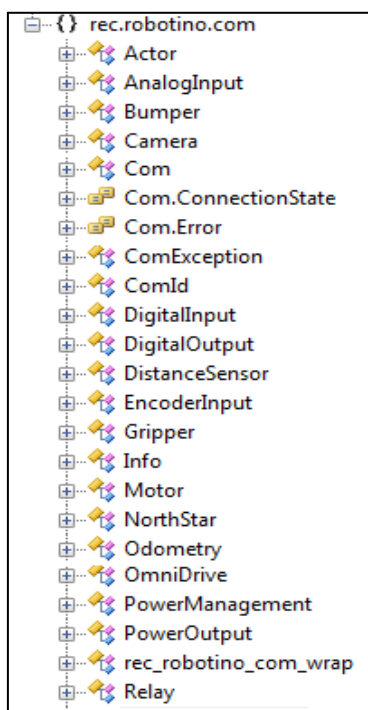


Figura 3.20: Librería *rec,robotino.com* y sus respectivas clases

Fuente: Autores

A continuación se analizan algunas de las clases y métodos que contiene la librería *rec.robotino.com* para el control del Robotino.

➤ Clase **AnalogInput**

Permite obtener valores de tensión usando los sensores inductivos u otros sensores de tipo análogo conectado a la regleta de conexiones.

MÉTODOS		
NOMBRE	DESCRIPCIÓN	PARÁMETROS
numAnalogInput()	Devuelve la entrada análoga que se está utilizando.	Ninguno
setInputNumber(uint)	Define la entrada análoga a utilizar.	Valores enteros (0 - 7)
value()	Devuelve un valor flotante del estado del sensor (0 – 10V).	Ninguno

Tabla 3.1: Clase *AnalogInput*

Fuente: Autores

➤ Clase Bumper

Utilizado para la detección de colisiones del Robotino con algún objeto en el entorno.

MÉTODOS		
NOMBRE	DESCRIPCIÓN	PARÁMETROS
Value ()	Devuelve un valor booleano (true/false).	Ninguno

Tabla 3.2: Clase Bumper
Fuente: Autores

➤ Clase Camera

Obtiene la imagen en tiempo real desde la webcam del Robotino con un tamaño y resolución personalizada.

MÉTODOS		
NOMBRE	DESCRIPCIÓN	PARÁMETROS
imageReceivedEvent(System.Drawing.Image data, uint dataSize, uint width, uint height, uint numChannels, uint bitsPerChannel, uint step)	Evento utilizado para actualizar las imágenes capturadas por la Webcam del Robotino.	Imagen, tamaño de datos del búfer jpeg, ancho, alto, número de canales, número de bit por canal, número de byte por cada fila de la imagen
isStreaming()	Devuelve el estado de la transmisión de la imagen.	Ninguno
setStreaming(bool streaming)	Activa o desactiva la transmisión de la imagen.	True/False
setResolution (uint width, uint height)	Define el tamaño de la imagen a capturar.	Ancho, alto

Tabla 3.3: Clase Camara
Fuente: Autores

➤ Clase Com

Establece la conexión entre la PC y el Robotino utilizando una dirección IP.

MÉTODOS		
NOMBRE	DESCRIPCIÓN	PARÁMETROS
address()	Devuelve un string con la dirección IP del Robotino.	Ninguno
connect()	Realiza la conexión con el Robotino.	Ninguno
Disconnect()	Deshabilita la conexión con el Robotino.	Ninguno
isConnected()	Devuelve el estado de la conexión.	Ninguno
setAddress(string address)	Define la IP del Robotino a utilizar.	Dirección IP

Tabla 3.4: Clase Com
Fuente: Autores

➤ Clase DigitalInput

Obtiene el estado (true o false) de los sensores ópticos u otros sensores conectados a las entradas digitales de la regleta de conexiones.

MÉTODOS		
NOMBRE	DESCRIPCIÓN	PARÁMETROS
numDigitalInputs()	Devuelve la entrada digital utilizada.	Ninguno
setInputNumber(uint n)	Define la entrada digital.	Valores enteros (0 - 7)
value()	Devuelve un valor binario (0 o 1).	Ninguno

Tabla 3.5: Clase DigitalInput
Fuente: Autores

➤ Clase DigitalOutput

Define las salidas digitales a utilizar en la regleta de conexiones del Robotino.

MÉTODOS		
NOMBRE	DESCRIPCIÓN	PARÁMETROS
numDigitalOutputs()	Devuelve el número de la salida digital (0 - 7).	Ninguno

setOutputNumber(uint)	Define la salida digital.	Valores enteros (0 - 7)
setValue(bool)	Establece el valor de salida.	True/False

Tabla 3.6: Clase DigitalOutput

Fuente: autores

➤ Clase DistanceSensor

Obtiene los valores de tensión y grados de ubicación de cada sensor de distancia.

MÉTODOS		
NOMBRE	DESCRIPCIÓN	PARÁMETROS
numDistanceSensors()	Devuelve el número del sensor de distancia (0 -8)	Ninguno
setSensorNumber (<u>uint</u> n)	Define el número del sensor de distancia a utilizar.	Valores enteros (0 - 8)
voltage()	Devuelve un valor del voltaje (0 – 2.5).	Ninguno

Tabla 3.7: Clase DistanceSensor

Fuente: Autores

➤ Clase EncoderInput

Recibe los valores del encoder incremental producidos por el desplazamiento del Robotino.

MÉTODOS		
NOMBRE	DESCRIPCIÓN	PARÁMETROS
position()	Devuelve un valor entero de la posición del Robotino.	Ninguno
resetPositionv()	Inicializa a cero los valores del encoder.	Ninguno
velocity()	Devuelve un entero con la velocidad del Robotino.	Ninguno

Tabla 3.8: Clase EncoderInput

Fuente: Autores

➤ Clase Motor

Esta clase permite controlar tanto el sentido de giro de los motores DC, velocidad de cada motor, frenado de los motores y un control PID en el caso de ser necesario.

MÉTODOS		
NOMBRE	DESCRIPCIÓN	PARÁMETROS
actualPosition()	Devuelve la posición actual del motor.	Ninguno
actualVelocity()	Devuelve la velocidad actual del motor.	Ninguno
motorCurrent()	Devuelve la corriente consumida por el motor.	Ninguno
resetPosition()	Inicializa a cero la posición del motor.	Ninguno
setBrake(bool)	Realiza un frenado en el motor.	True/False
setMotorNumber (<u>uint</u> number)	Definir el número del motor a utilizar	Valores enteros (0 - 2)
setPID(<u>byte</u> kp, <u>byte</u> ki, <u>byte</u> kd)	Establece las constantes del controlador PID.	Constante proporcional, constante integral, constante diferencial.
setSpeedSetPoint(<u>float</u> speed)	Define la velocidad del motor en rpm.	Valores enteros (0 - 1500)

Tabla 3.9: Clase Motor

Fuente: Autores

➤ Clase OmniDrive

Calcula las velocidades de cada motor para realizar desplazamientos en dirección 'x', 'y' y omega.

MÉTODOS		
NOMBRE	DESCRIPCIÓN	PARÁMETROS
project (out <u>float</u> m1, out <u>float</u> m2, out <u>float</u> m3, <u>float</u> vx, <u>float</u> vy, <u>float</u> omega)	Define las velocidades únicas de los motores para trabajar con coordenadas cartesianas.	Velocidad motor1, velocidad motor2, velocidad motor3, en x, velocidad en y,

		velocidad en omega
setVelocity(float vx, float vy, float omega)	Define las velocidades para las coordenadas x, y, omega.	Velocidad en x, velocidad en y, velocidad en omega

Tabla 3.10: Clase OmniDrive
Fuente: Autores

3.1.3 PRUEBAS DE LOS SENSORES

En este apartado se muestran las pruebas y análisis de los sensores más importantes utilizando el robot móvil Robotino de FESTO, con el fin de cumplir los objetivos planteados en el presente proyecto de titulación, los cuales se mencionan a continuación.

Sensor óptico: Se realizaron las pruebas necesarias con materiales de diferentes colores, con el fin de observar el comportamiento del sensor óptico en cada una de ellas. Estas pruebas se pueden apreciar en la tabla 3.11.

Color	Estado del sensor 1	Estado del sensor 2
Blanco	False	False
Negro	True	True
Azul	False	False
Rojo	False	False
Verde	False	False
Marrón	False	False

Tabla 3.11: Comportamiento del sensor óptico frente a materiales de diferente color
Fuente: Autores

Sensor inductivo: Para observar su comportamiento se realizaron las pruebas utilizando diferentes tipos de materiales metálicos, los resultados de cada material se muestran en la tabla 3.12.

Material	Tensión de salida (V)	Magnético	Distancia (mm)
Madera	9,96	No	2
Aluminio	6,83	No	2

Cobre	6,56	No	2
Latón	4,49	Si	2
Hierro	2,77	Si	2

Tabla 3.12: Tensión de salida del sensor inductivo ante diversos materiales

Fuente: Autores

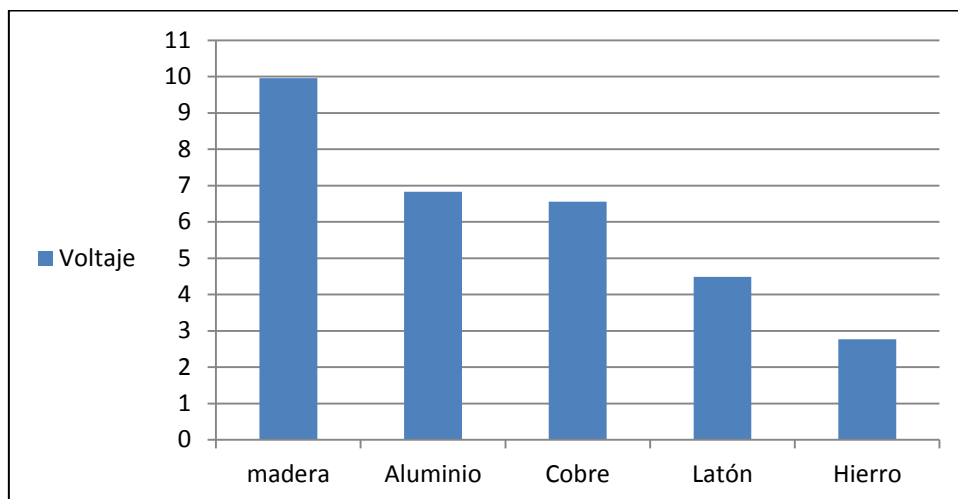


Figura 3.21: Tensión de salida del sensor inductivo ante diversos materiales

Fuente: Autores

Webcam: Se realizaron las pruebas capturando una imagen desde una aplicación desarrollada bajo C# .NET, con el fin de obtener una imagen nítida, y determinar la distancia más adecuada para colocar el código de barras para su identificación. Las pruebas se muestran en la figura 3.22.

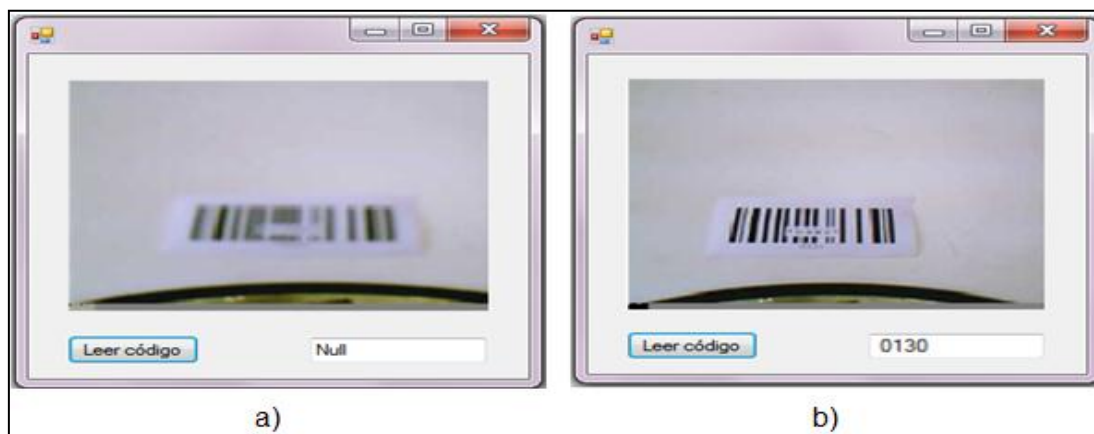


Figura 3.22: a) Imagen fuera de foco b) Imagen enfocada

Fuente: Autores

3.2 DISEÑO

3.2.1 DIAGRAMAS DE CASOS DE USO

Para representar el funcionamiento del sistema, se ha elegido la herramienta UML (Lenguaje Unificado de Modelado), ya que dispone de un conjunto de herramientas que permiten modelar sistemas orientados a objetos debido a su presentación gráfica y explicativa. En los siguientes diagramas se muestran los puntos más importantes sobre el diseño del sistema.

1. Creación del entorno virtual
2. Control manual
3. Configuración de parámetros del Robotino
4. Configuración inicial del algoritmo
5. Navegación del Robotino

Creación del entorno virtual

En este caso de uso el usuario inicia creando la dimensión del área de trabajo, dentro de la misma, el sistema permite al usuario agregar y eliminar nodos y arcos que representarán los estados y caminos por donde se desplazará el Robotino.

En este caso de uso también el usuario puede configurar los parámetros para la identificación de cada camino que son utilizados para desplazarse desde un estado a otro.

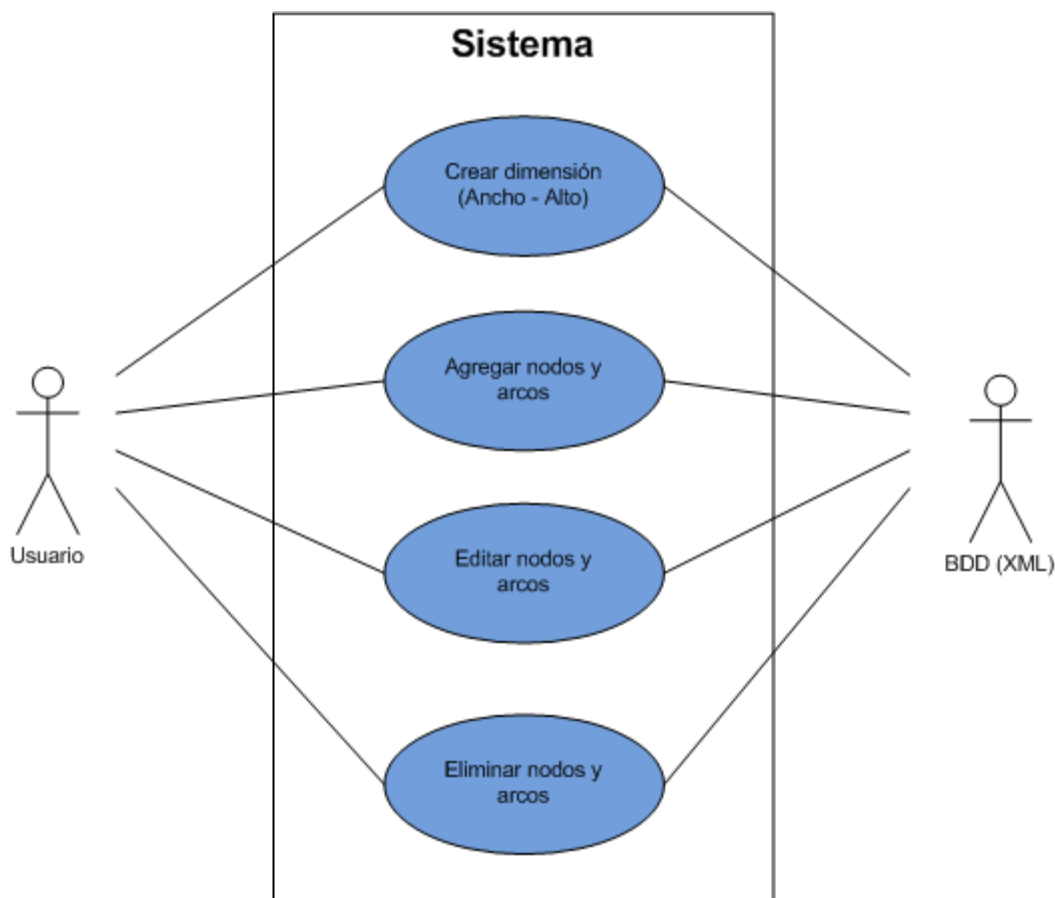


Figura 3.23: Creación del entorno virtual
Fuente: Autores

Control manual

El usuario antes de interactuar con el Robotino debe establecer una comunicación (PC → Robotino), ingresando una dirección IP que está asignada en él. Con esto se logra establecer un canal de comunicación WLAN de envío y recepción de datos si esta es ingresada correctamente. Además el usuario puede ejecutar el Robotino para que se desplace en modo autónomo, o también puede detenerlo en el caso de ser necesario.

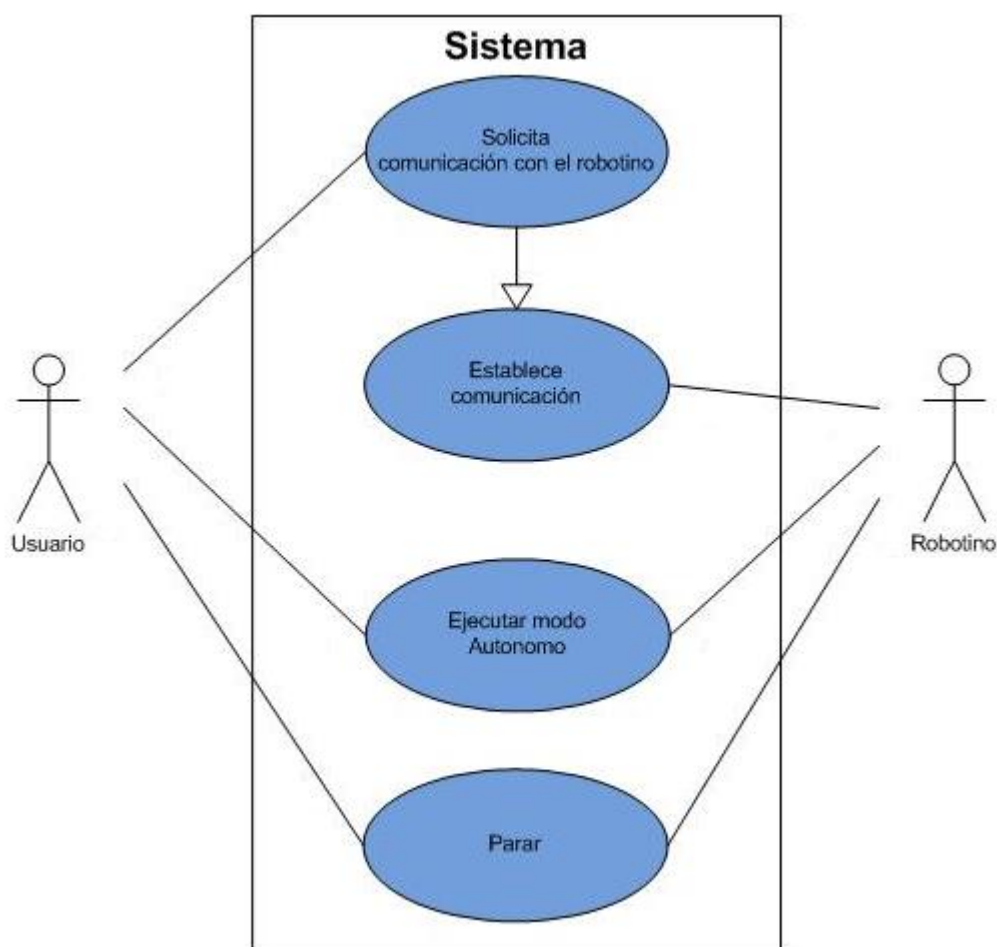


Figura 3.24: Control manual

Fuente: Autores

Configuración de parámetros del Robotino

El usuario tendrá la posibilidad de modificar ciertos parámetros del Robotino antes de ejecutar el proceso de búsqueda, tales como: límite interior y exterior del material utilizado para el seguidor inductivo, velocidades de cada motor desde 0 a 600 RPM, como también seleccionar entradas analógicas y digitales para la lectura de los datos provenientes de los sensores que posee el Robotino.

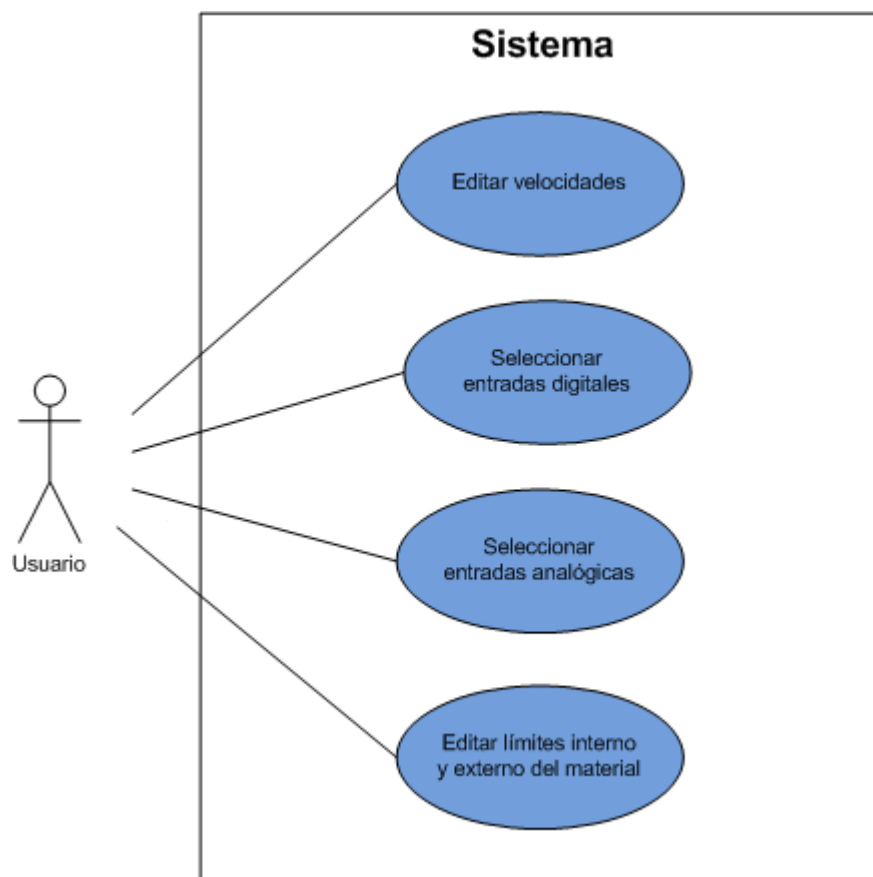


Figura 3.25: Configuración de parámetros del Robotino
Fuente: Autores

Configuración inicial del algoritmo

Para que el algoritmo inicie su funcionamiento se requieren ciertos parámetros que el usuario debe ingresar, tales como: nodo inicio y nodo meta. Estos parámetros también son utilizados para generar las distancias en línea recta (DLR) desde el nodo meta hacia todos los nodos que forman parte del grafo.

DLR o también denominado $h(n)$, es utilizado para calcular la función de evaluación $f(n)$ en el proceso de búsqueda.

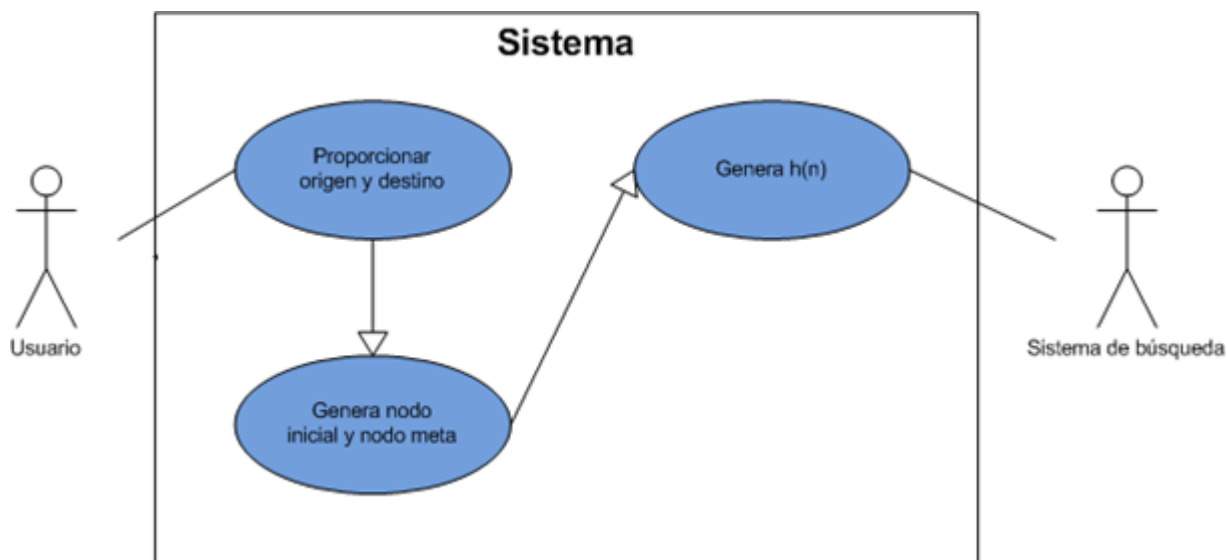


Figura 3.26: Configuración inicial del algoritmo

Fuente: Autores

Navegación del Robotino

El usuario empieza activando el algoritmo de búsqueda para explorar el grafo creado, y al mismo tiempo el Robotino comienza ejecutando una secuencia de acciones controladas por el sistema para identificar todos los caminos generados por el algoritmo, con el fin de llevar al Robotino al estado meta. Una vez alcanzado el estado meta, el sistema detiene al Robotino finalizando la búsqueda. Además, el usuario también tiene la posibilidad de detener todo el sistema de búsqueda en caso de ocurrir algún imprevisto con el móvil.

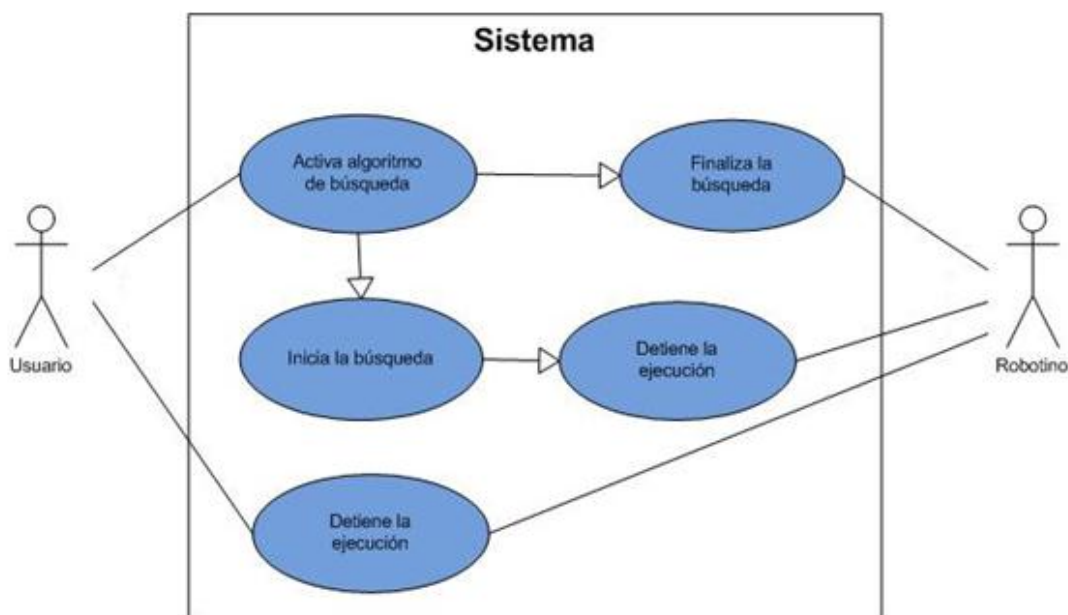


Figura 3.27: Navegación del Robotino
Fuente: Autores

3.2.2 DISEÑO DEL ENTORNO DE TRABAJO

Para la demostración de este proyecto de tesis se ha diseñado un grafo formado por 15 nodos y 25 arcos enlazados entre sí sobre un plano cartesiano como se muestra en la figura 3.28, donde un nodo está representado por las coordenadas de un punto, que se forman asociando un valor del eje de las "X" y uno de las "Y". La distancia de cada arco se calcula haciendo uso del teorema de Pitágoras.

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Dónde:

x_1 y y_1 → Son las coordenadas del punto 1 (nodo n)

x_2 y y_2 → Son las coordenadas del punto 2 (nodo n1)

d → Es la distancia obtenida entre el punto 1 y punto 2 (Coste $g(n)$)

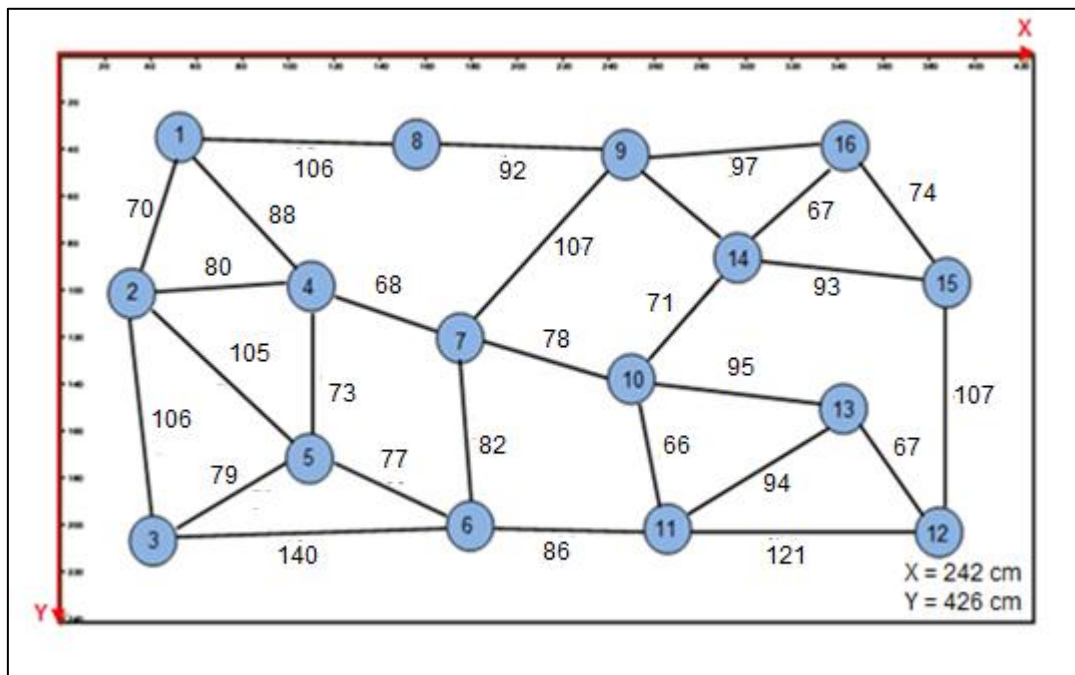


Figura 3.28: Diseño del entorno de trabajo
Fuente: Autores

Cálculo de la distancia entre dos nodos

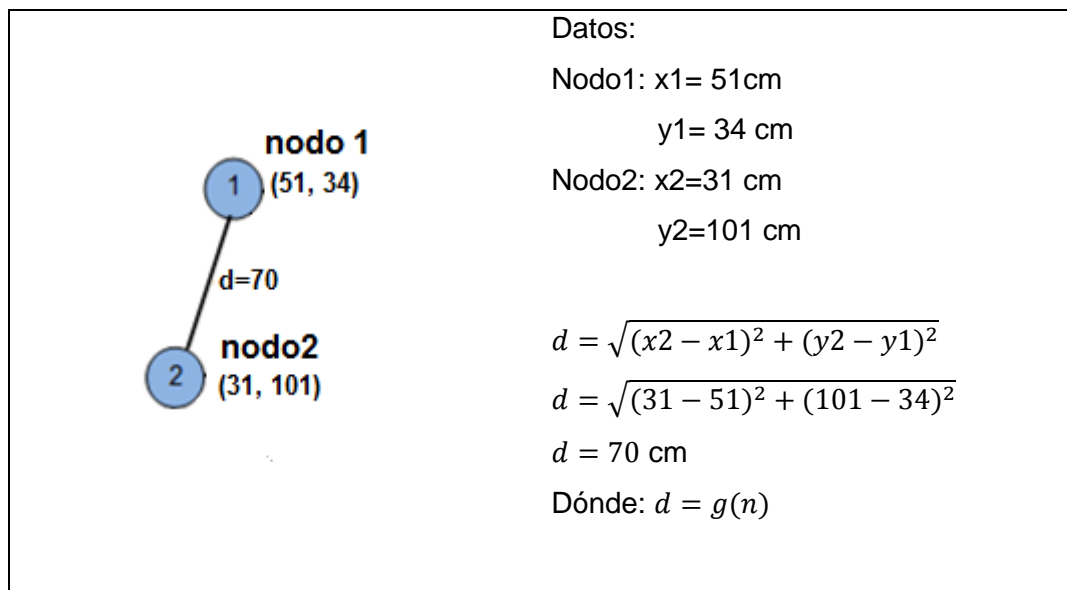


Figura 3.29: Cálculo de la distancia entre dos nodos
Fuente: Autores

3.2.3 ARQUITECTURA DEL SISTEMA DE CONTROL

Para este proyecto de tesis se ha seleccionado la arquitectura de control deliberativa para una navegación más eficiente.

En la figura 3.30 se muestra la arquitectura deliberativa compuesta por dos niveles (superior e inferior). El nivel superior de la arquitectura se encuentra en una estación de trabajo bajo Windows XP, permitiendo la toma de decisiones mediante el empleo del módulo de Inteligencia Artificial (IA) desarrollado bajo la plataforma C# .NET. Las órdenes generadas en este nivel son enviadas al nivel inferior de la arquitectura a través de una comunicación WLAN para ejecutar las acciones en el Robotino, utilizando un sistema operativo basado en GNU/Linux instalado dentro de él.

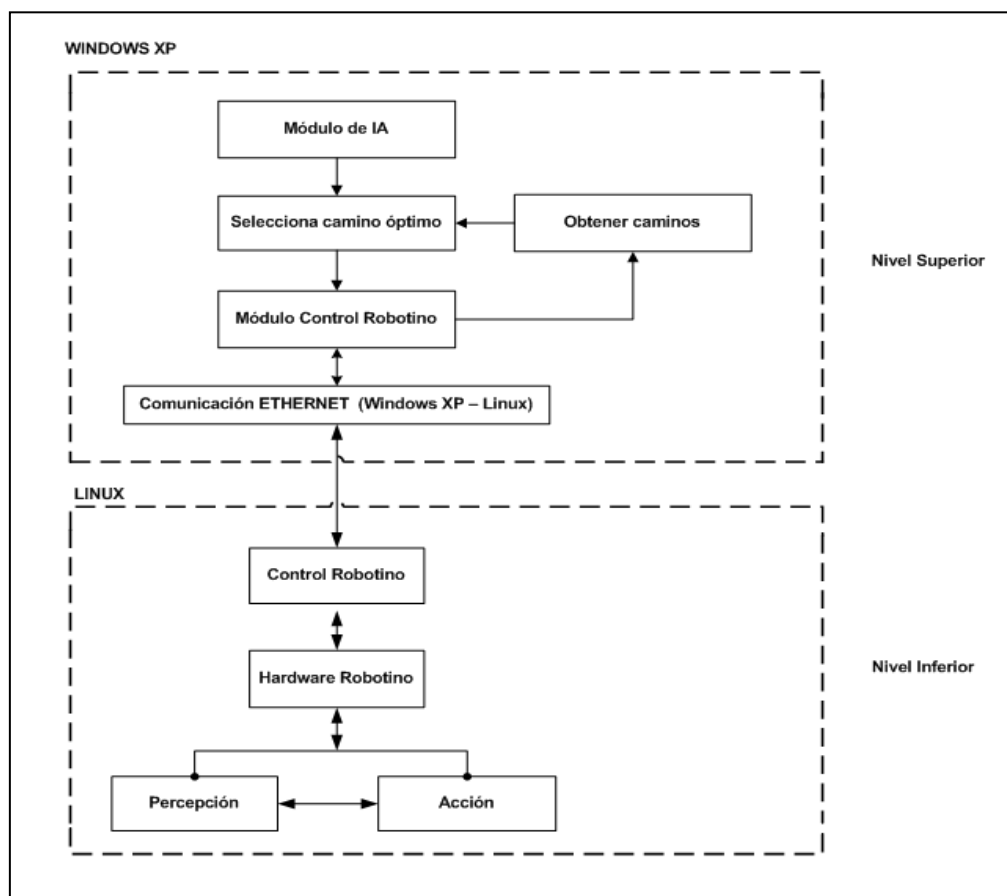
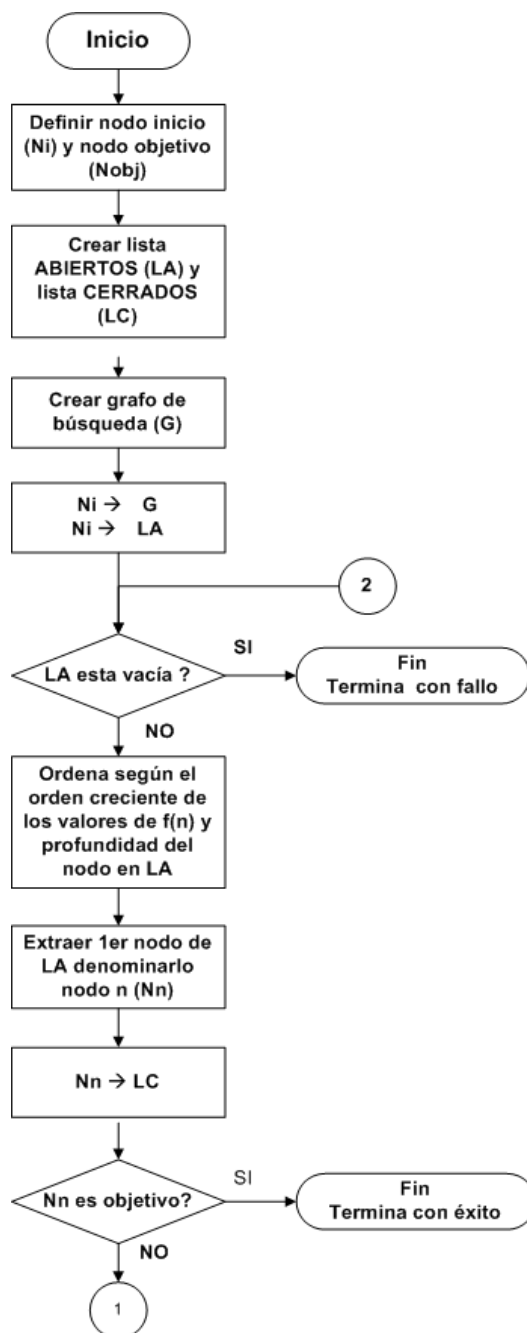


Figura 3.30: Arquitectura de Control Deliberativa para Robotino de FESTO

Fuente: Autores

3.3 DISEÑO DEL ALGORITMO

El diagrama de flujo presentado a continuación muestra una representación gráfica del algoritmo de búsqueda informada A*, con el fin de comprender de mejor manera el funcionamiento de los procesos que se realizan en él para encontrar el camino más óptimo en un grafo



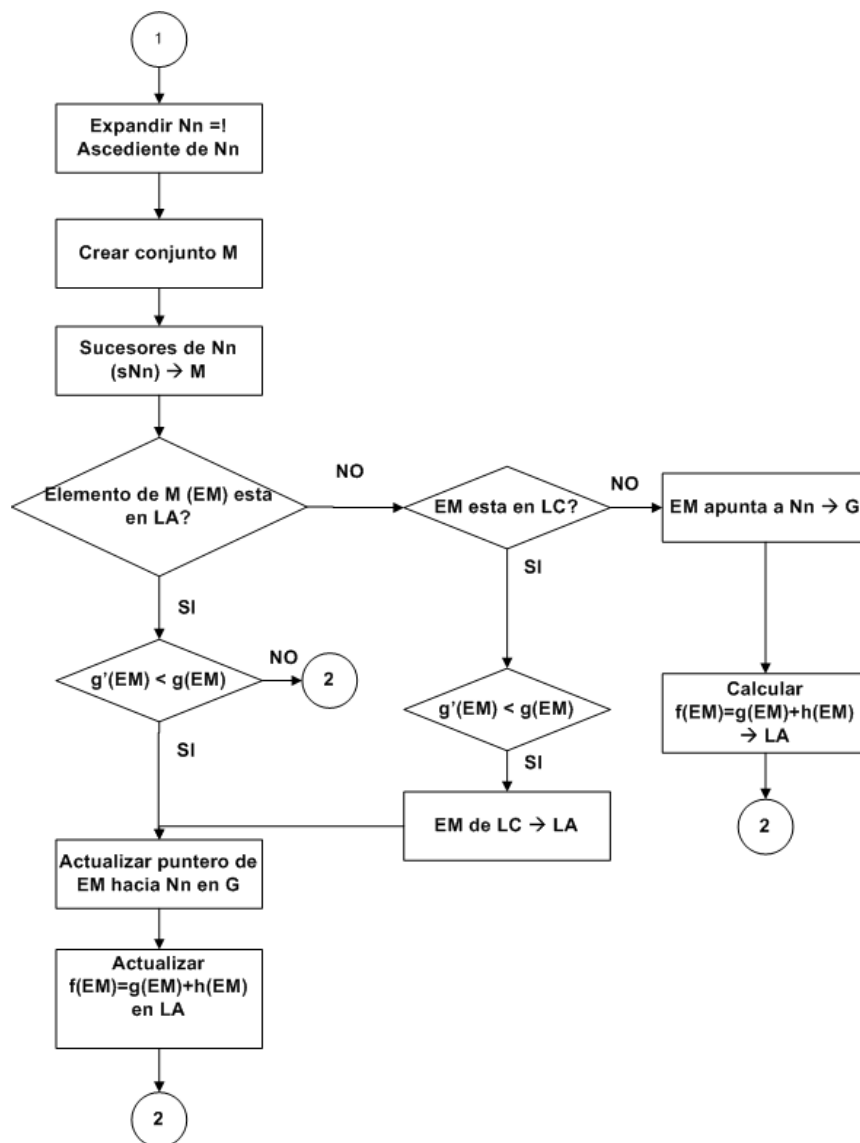


Figura 3.31: Proceso A*

Fuente: Autores

3.4 INTERACCIÓN CON EL ENTORNO Y CARACTERÍSTICAS EN TIEMPO REAL

Para realizar la búsqueda en tiempo real es necesario que el Robotino ejecute una secuencia de tareas que le permitan al mismo llegar a su objetivo de forma autónoma. En los siguientes diagramas de flujo se muestra un análisis de las tareas

que ejecuta el Robotino para encontrar el camino más óptimo dentro de un entorno controlado, utilizando la plataforma C# .NET para el control total de sus movimientos.

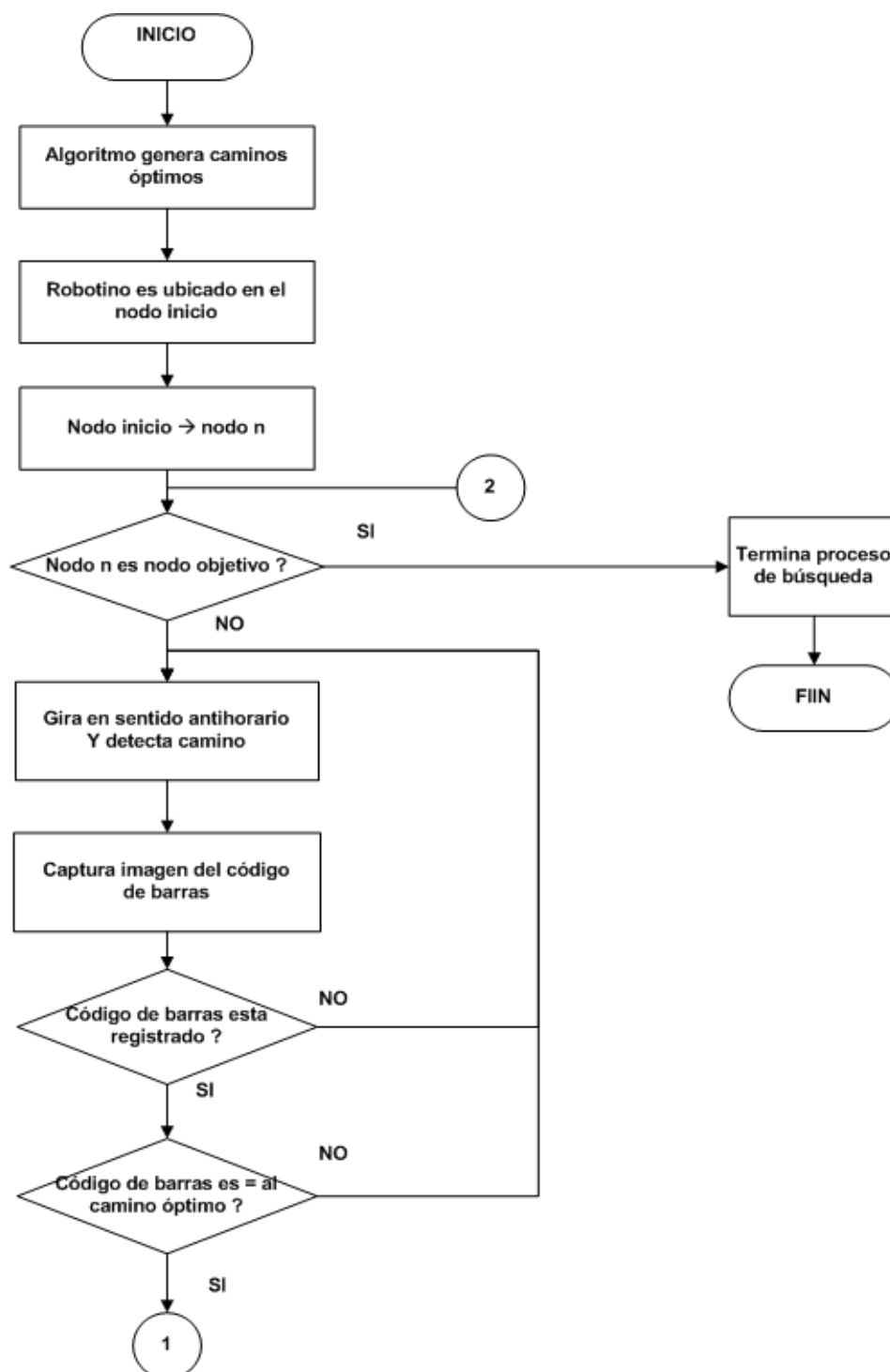


Figura 3.32: Identificación del nodo objetivo
Fuente: Autores

El diagrama muestra la manera como el Robotino identifica los posibles caminos y nodos dentro del entorno. El funcionamiento empieza situándose en el nodo inicial y termina su proceso solamente cuando se situé en el nodo objetivo.

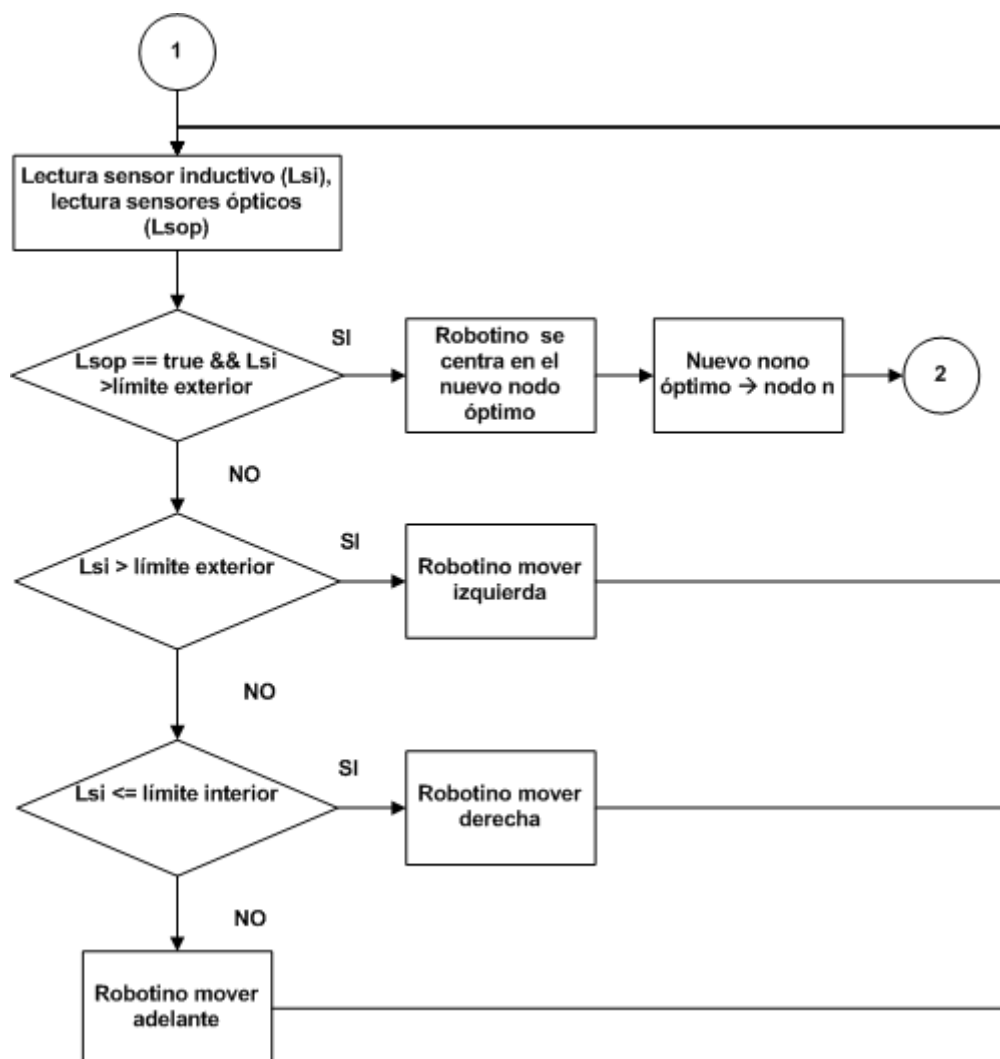


Figura 3.33: Seguidor inductivo
Fuente: Autores

El diagrama muestra la tarea de seguimiento de camino que realiza el Robotino para desplazarse desde un nodo a otro utilizando un sensor inductivo para detectar el material metálico. La tarea finaliza cuando los sensores ópticos cambian de estado y el sensor inductivo está fuera del material.

Los límites son utilizados para definir las condiciones del seguidor inductivo, es decir, cuando el Robotino tiene que moverse a la derecha, izquierda o hacia adelante.

La figura 3.34 muestra los límites utilizados como referencia para el desplazamiento del Robotino. El límite interior se observa en el medio del material metálico, ya que solo ahí se puede evaluar el valor puro del mismo y el Robotino tiende a desviarse hacia la derecha, mientras que el límite exterior está dado por un material no metálico (madera) que lo ayudará a desviarse hacia la izquierda. Hay que tomar en cuenta que el Robotino se moverá en línea recta solamente cuando se encuentra sobre el material metálico y no metálico, ya que solo ahí se produce una división de voltaje que ayudará al móvil a funcionar como un seguidor tradicional.

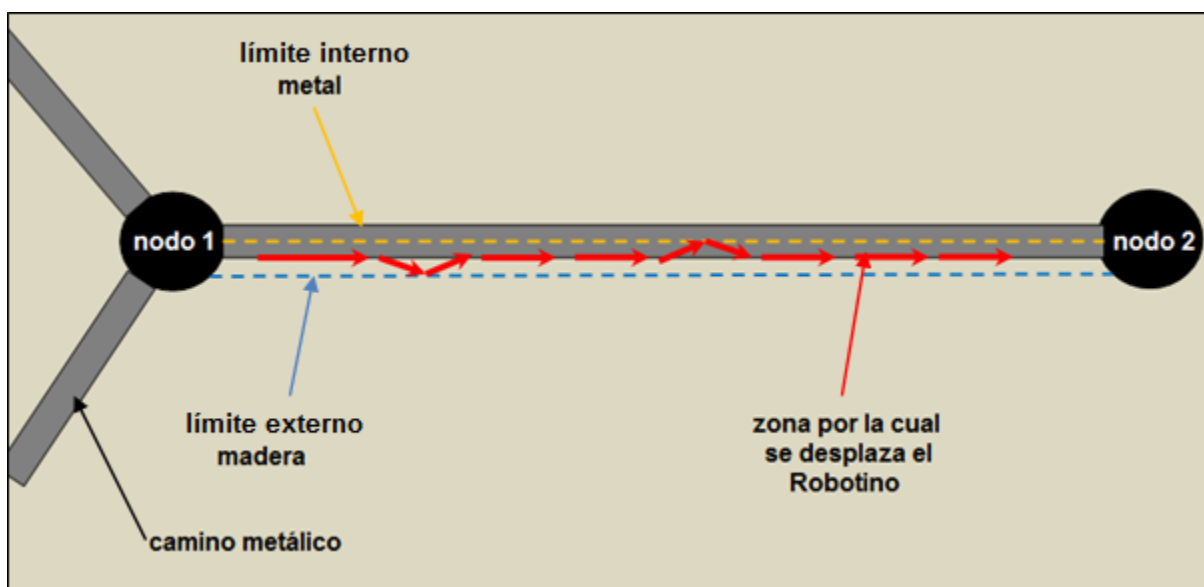


Figura 3.34: Límite interno y externo del camino
Fuente: Autores

3.5 DISEÑO DE LA INTERFAZ

Las siguientes figuras muestran un prototipo del sistema de búsqueda, diseñado en base a los parámetros requeridos durante el análisis.

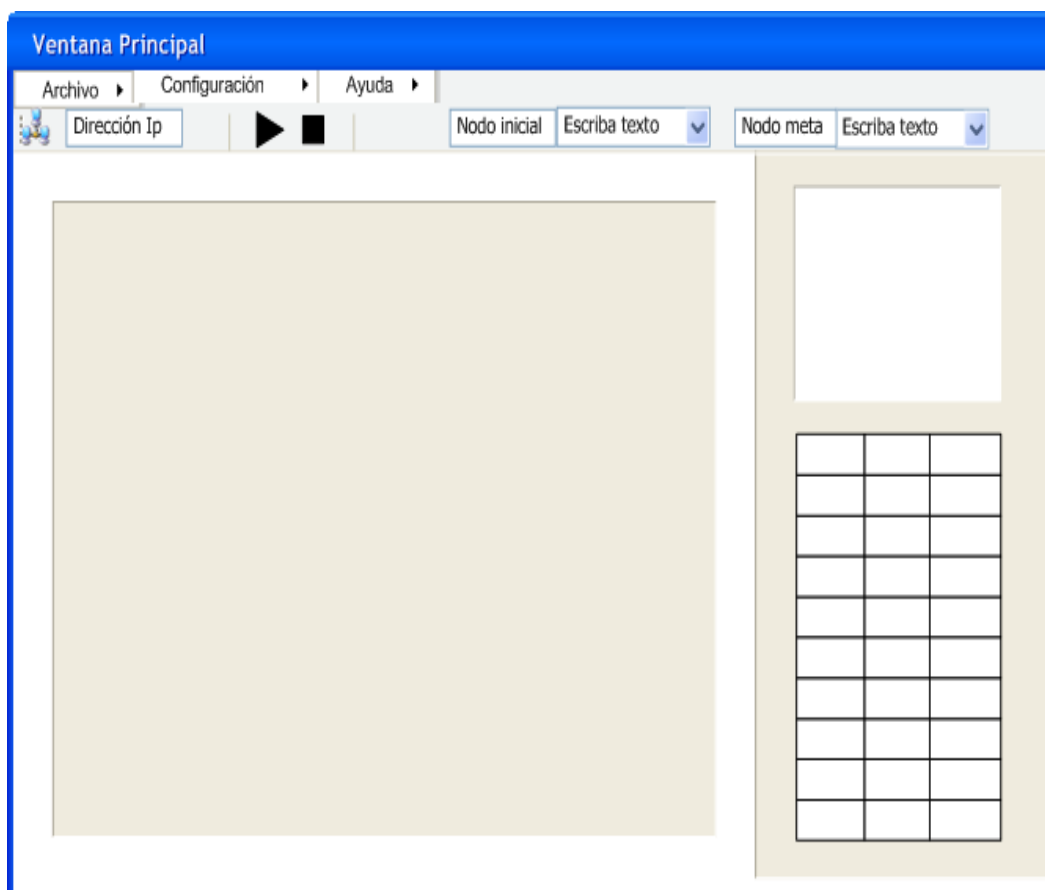


Figura 3.35: Ventana principal
Fuente: Autores

El sistema está compuesto por una ventana principal como se muestra en la figura 3.36 donde se visualiza el proceso de búsqueda, además la misma ventana permite gestionar los parámetros necesarios para su funcionamiento.

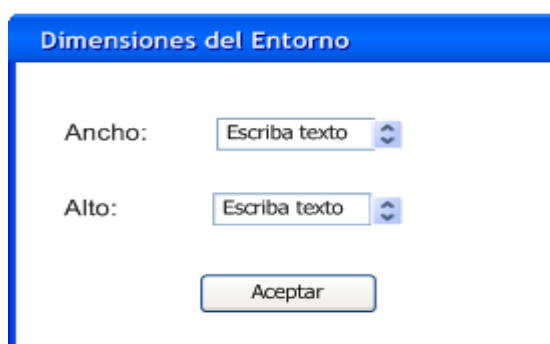


Figura 3.36: Ventana de creación del entorno de trabajo
Fuente: Autores

La figura 3.37 define los parámetros iniciales para la creación del entorno virtual, donde las dimensiones del entorno están definidas en centímetros.

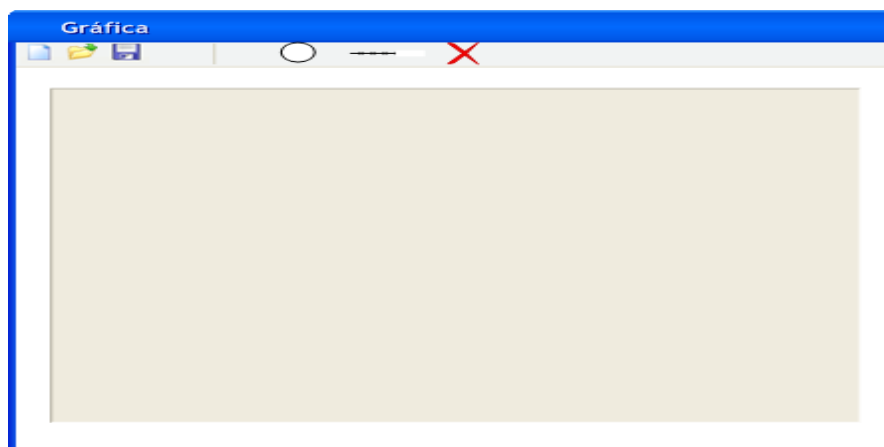


Figura 3.37: Ventana para crear el entorno
Fuente: Autores

En la figura 3.38 muestra las configuraciones iniciales del Robotino antes de empezar su funcionamiento de forma autónoma.

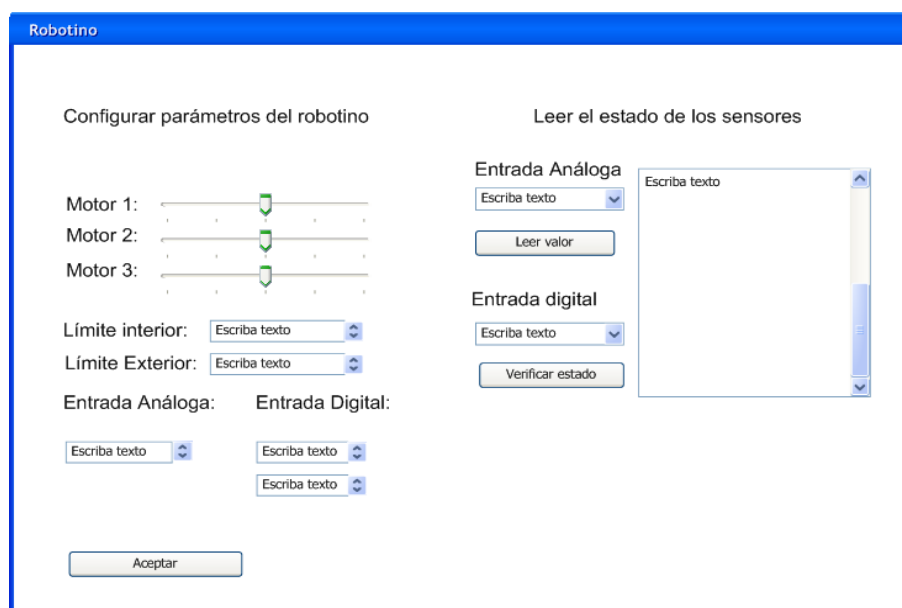


Figura 3.38: Ventana para la configuración de parámetros del Robotino
Fuente: Autores

CAPÍTULO 4

IMPLEMENTACIÓN Y PRUEBAS

4.1 DIAGRAMA DE CLASES

El sistema está compuesto por diferentes clases, la interacción de todas ellas permiten cumplir los objetivos planteados en este proyecto de tesis. El menú principal realiza las llamadas necesarias de las clases con el fin de presentar visualmente los resultados obtenidos durante la ejecución del sistema de búsqueda. Cada clase realiza una función específica, lo que le permite ser reutilizable en otras aplicaciones.

La figura 4.1 muestra las clases implementadas en el presente proyecto de tesis para encontrar el camino más óptimo utilizando el robot móvil Robotino de FESTO. Cada clase se encuentra marcada por un color diferente. Estos son detallados en la tabla 4.1.




Color	Descripción
	Clases que permiten interactuar al usuario con el sistema de búsqueda.
	Clases de control, encargadas de realizar los cálculos del algoritmo y el control del Robotino.
	Clases externas utilizadas mediante referencias en sistema de búsqueda.

Tabla 4.1: Descripción de clases
Fuente: Autores

4.2 PROGRAMACIÓN DE SOFTWARE

En este apartado se explica cómo se dio solución a lo planteado en el capítulo 3, con el fin de ver de una manera clara y concisa el funcionamiento del sistema de búsqueda informada. El sistema está compuesto por 5 módulos, tomando en cuenta que la conexión de todos ellos da forma a la totalidad del proyecto, siendo estos:

- **Módulo gráfico:** Este módulo permite crear el entorno virtual en el frame principal, proporcionando la información necesaria para el proceso de búsqueda.
- **Módulo barcodeLib:** Se encarga del procesamiento y reconocimiento del código de barras.
- **Módulo de datos:** Este módulo es encargado de gestionar la información necesaria para el funcionamiento del algoritmo de Inteligencia Artificial.
- **Módulo de Inteligencia Artificial (IA):** Permite tomar las decisiones y elegir el camino más óptimo a lo largo del proceso de búsqueda utilizando la información del módulo de datos.
- **Módulo Robotino:** Finalmente, este módulo es el encargado de ejecutar las acciones necesarias para llevar a cabo la tarea de búsqueda en el entorno real.

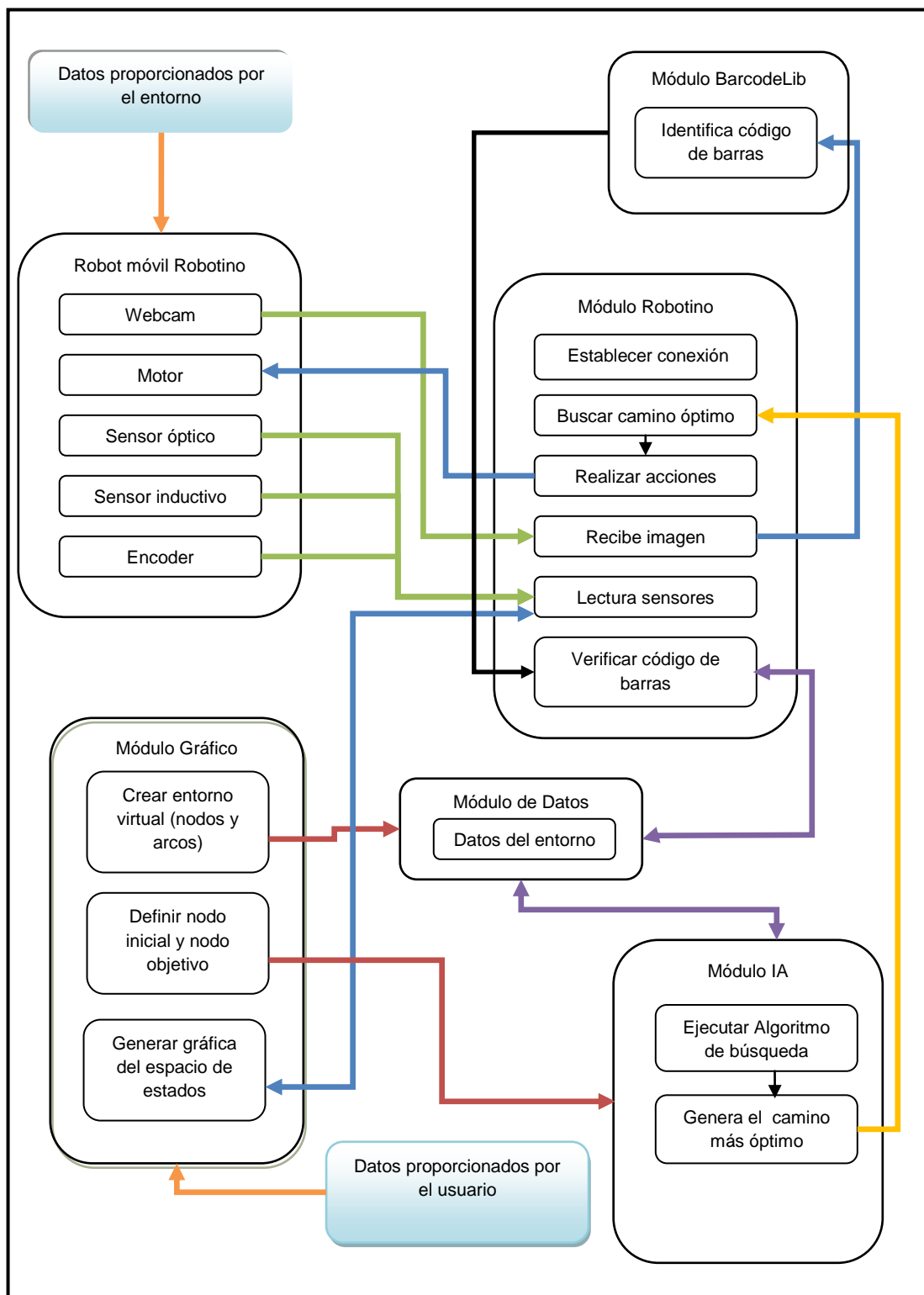


Figura 4.2: Esquema del funcionamiento final del Sistema de Búsqueda
Fuente: Autores

Módulo Gráfico

- **Entrada:** La entrada de este módulo es simplemente la posición “x” y “y” de cada nodo creado en el entorno virtual.
- **Proceso:** Este módulo se encarga de crear una representación gráfica en 2D de los nodos y arcos dentro del entorno virtual.

El coste de cada arco $g(n)$ creado en el entorno virtual es calculado utilizando el teorema de Pitágoras. Este valor será útil cuando el módulo de Inteligencia Artificial calcule la función de evaluación $f(n)$ de cada nodo. En la figura 4.3 se muestra como el módulo calcula el coste de uno de los arcos.

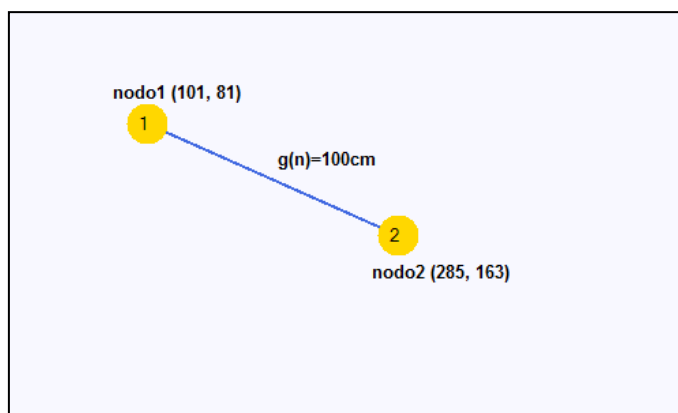


Figura 4.3: Cálculo del coste
Fuente: Autores

Cálculo del coste ($g(n)$)

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

$$d = \sqrt{(285 - 101)^2 + (163 - 81)^2}$$

$$d = 201,44 \text{ pixeles}$$

Conversión de pixeles a cm

$$g(n) = \text{Math.Celling}(d);$$

$$g(n) = 100 \text{ cm, } 2 \text{ pixeles equivalen a } 1 \text{ cm}$$

Las posiciones de los nodos y arcos son almacenados en el módulo de datos, siendo útiles tanto para la simulación virtual del sistema de búsqueda como para la demostración de la misma utilizando el Robotino.

- **Salida:** Como salida de este módulo se obtiene un imagen de tipo bitmap, permitiendo visualizar en el entorno virtual la representación del grafo creado por el usuario y el espacio de estados generado durante el proceso de búsqueda.

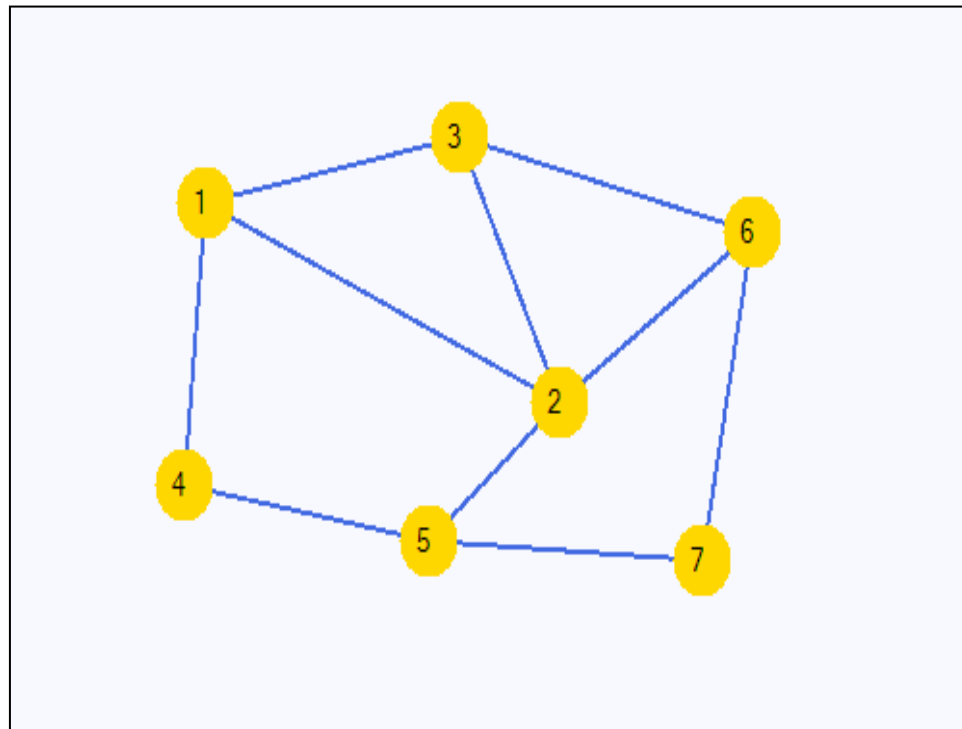


Figura 4.4: Imagen del entorno virtual tipo bitmap
Fuente: Autores

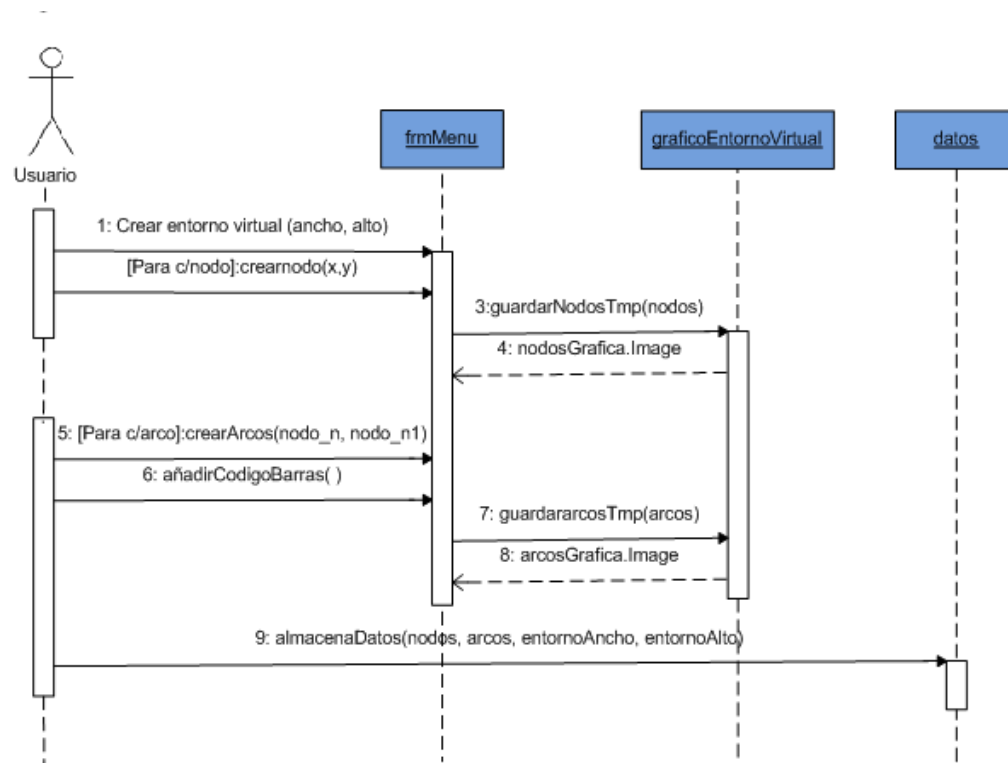


Figura 4.5: Diagrama de secuencia del Módulo Gráfico

Fuente: Autores

Módulo BarcodeLib:

- **Entrada:** Este módulo tiene como entrada la imagen capturada por la webcam del Robotino.
- **Proceso:** La imagen capturada por la webcam del Robotino es enviada a este módulo con el fin de obtener la información necesaria que nos permita identificar los caminos que se encuentra alrededor de un nodo en el entorno de trabajo. Sin embargo, no se sabe con certeza cómo realiza el tratamiento de la imagen, tan solo se sabe que es un módulo ya diseñado y nos proporciona los datos que se necesitan y que son vitales en la toma de decisiones para realizar acciones sobre el Robotino.

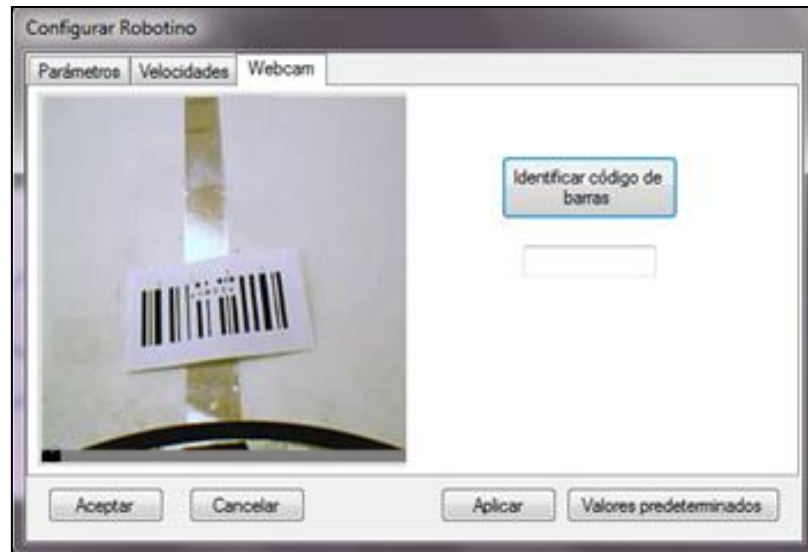


Figura 4.6: Código de barras capturado por la webcam del Robotino
Fuente: Autores

- **Salida:** Este módulo nos proporciona un String con la identificación del código de barras contenido en la imagen capturada.



Figura 4.7: String con la identificación del código de barras
Fuente: Autores

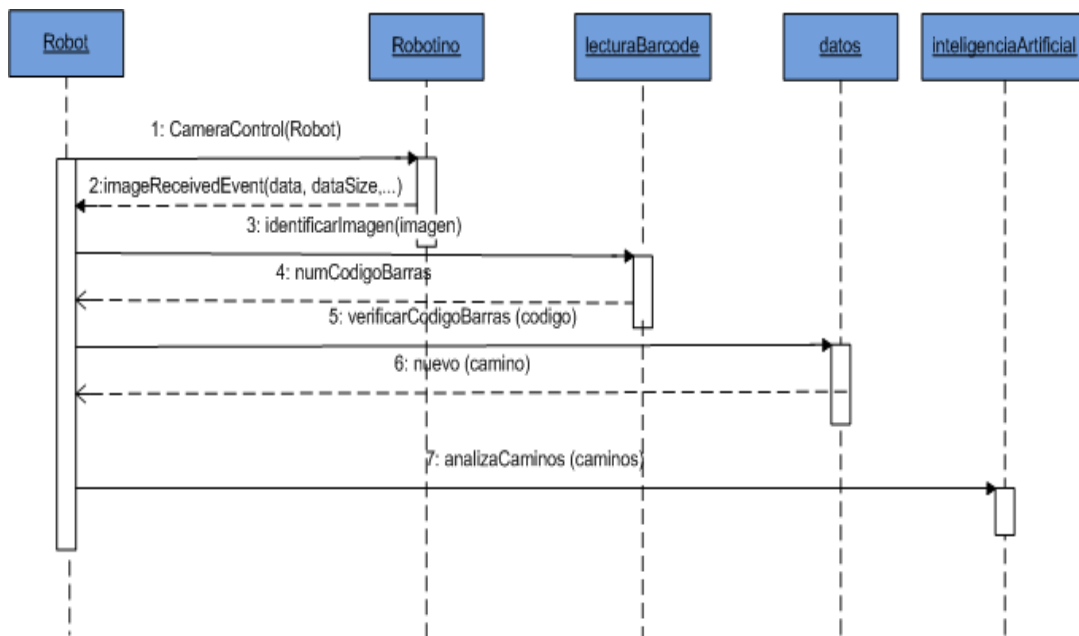


Figura 4.8: Diagrama de secuencia del Módulo BarcodeLib
Fuente: Autores

Módulo de datos

- **Entrada:** Las entradas para este módulo son las matrices que contienen la información de los arcos y nodos, como también las dimensiones del entorno virtual y configuraciones necesarias para utilizar el Robotino. La matriz de nodos contiene las posiciones “x” y “y”, mientras que la matriz de arcos contiene los enlaces entre los nodos, donde cada enlace tiene un coste y un código de barras.
- **Proceso:** Toda la información obtenida en las entradas del módulo es almacenada en un DataSet y enviada a un archivo XML, lo que le permite al sistema de búsqueda ser portable, sin la necesidad de realizar conexiones con bases de datos externas.

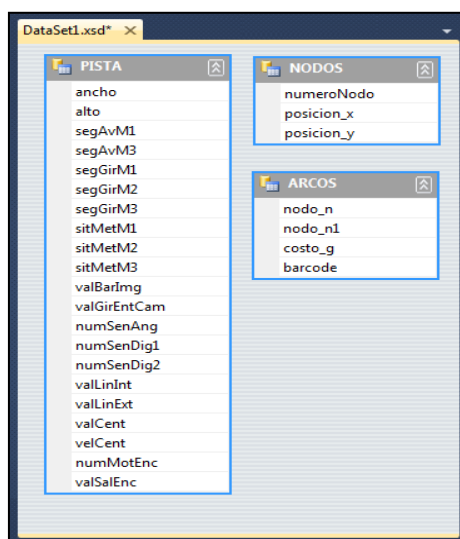


Figura 4.9: Tablas utilizadas para la gestión de la información del entorno virtual

Fuente: Autores

La figura 4.9 muestra las tablas utilizadas dentro de un dataSet para almacenar la información más importante del entorno virtual, siendo esta proporcionada por el usuario al crear un nuevo entorno en el sistema o a través de un archivo XML que contiene la información de un entorno ya creado.

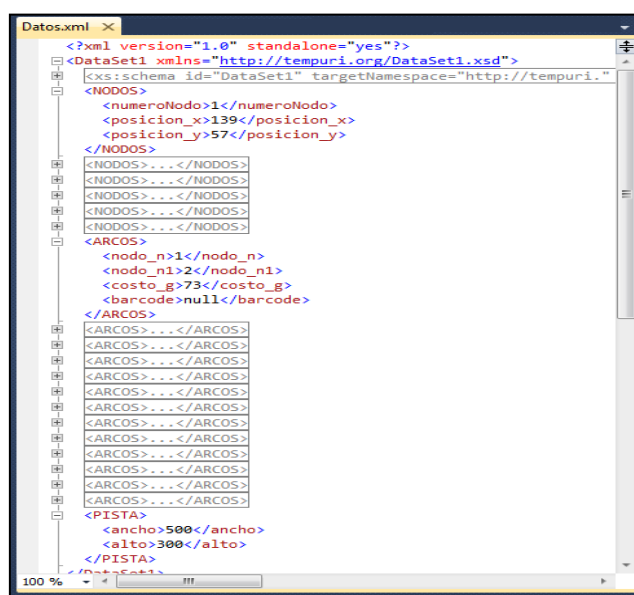


Figura 4.10: Archivo XML generado desde un dataSet por el Sistema de Búsqueda

Fuente: Autores

A continuación se muestra la estructura del documento XML utilizado para almacenar la información necesaria del sistema de búsqueda.

```
<?xml version="1.0" standalone="yes"?>
```

```
<NODOS>
```

```
<numeroNode> Número del nodo </numeroNode>
```

```
<posicion_x> Posición en X </posicion_x>
```

```
<posicion_y> Posición en Y </posicion_y>
```

```
</NODOS>
```

```
<ARCOS>
```

```
<nodo_n> Número del nodo n </nodo_n>
```

```
<nodo_n1> Número del nodo n1 </nodo_n1>
```

```
<coste_g> Coste del nodo n al nodo n1 </coste_g>
```

```
<barcode> Código de barras </barcode>
```

```
</ARCOS>
```

```
<PISTA>
```

```
<ancho> Ancho </ancho>
```

```
<alto> Alto </alto>
```

```
<segAvM1> Velocidad seguidor motor1 </segAvM1>
```

```
<segAvM3> Velocidad seguidor motor3 </segAvM3>
```

```
<segGirM1> Velocidad seguidor giro motor1 </segGirM1>
```

```
<segGirM2> Velocidad seguidor giro motor2 </segGirM2>
```

```
<segGirM3> Velocidad seguidor giro motor3 </segGirM3>
```

```
<valBarImg> Velocidad barrido código de barras </valBarImg>
```

```
<valGirEntCam> Velocidad giro entre caminos </valGirEntCam>
```

```
<numSenAng> Número de entrada analógica </numSenAng>
```

```
<numSenDig1> Número de entrada digital1 </numSenDig1>
```

```
<numSenDig2> Número de entrada digital2 </numSenDig2>
```

```
<valLinInt> Valor límite interior </valLinInt>
```

```
<valLinExt> Valor límite exterior </valLinExt>
```

<valCent> *Valor de centrado* </valCent>

<velCent> *Velocidad de centrado* </velCent>

<numMotEnc> *Motor de referencia para el centrado* </numMotEnc>

<valSalEnc> *Valor de salida del camino* </valSalEnc>

</PISTA>

La información del archivo XML puede ser cargada y editada en el Sistema de Búsqueda en las ocasiones que se considere necesario sin la necesidad de crear un entorno virtual desde cero.

- **Salida:** Como salida de este módulo se obtiene la información del entorno virtual que es utilizada para el funcionamiento de los demás módulos cuando el proceso de búsqueda entra en funcionamiento.

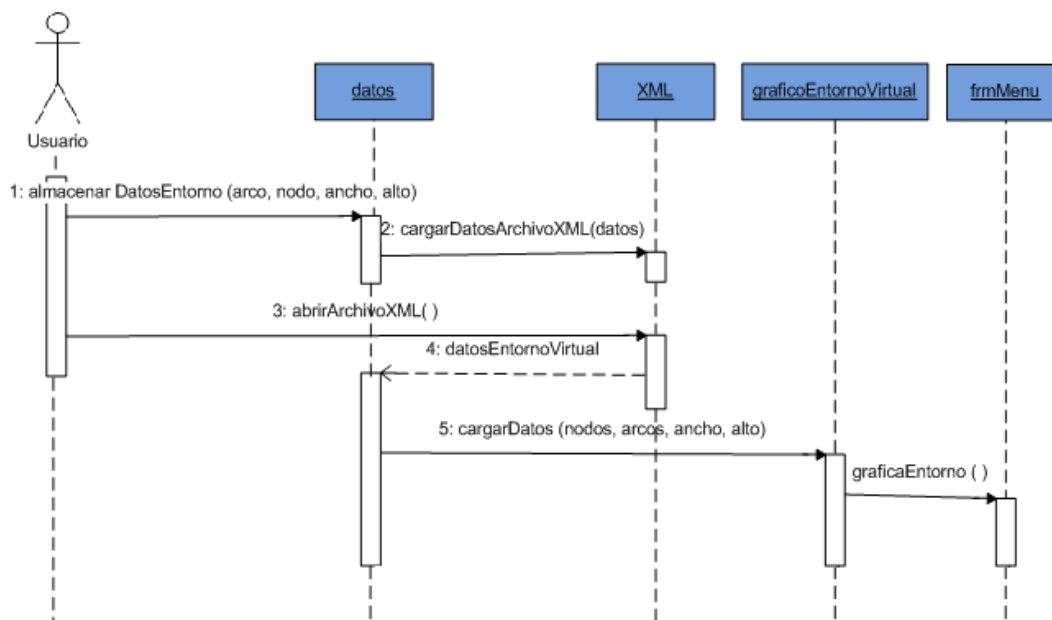


Figura 4.11: Diagrama de secuencia del Módulo de Datos

Fuente: Autores

Módulo Inteligencia Artificial (IA)

- **Entrada:** Este módulo tiene como entrada el nodo inicial, el nodo objetivo y el grafo de búsqueda. Estos datos son de vital importancia para inicializar el proceso de búsqueda del camino más óptimo.
- **Proceso:** El proceso que se lleva a cabo en este módulo es de vital importancia, ya que en este se ejecuta el algoritmo de búsqueda A* para generar el camino más óptimo.

Una vez proporcionadas las entradas, el módulo genera las distancias en línea recta para cada nodo utilizando el teorema de Pitágoras, es decir desde el nodo objetivo hacia los nodos que le rodean, donde el nodo objetivo siempre es cero. En la Figura 4.12 se muestra un ejemplo utilizando el sistema de búsqueda.

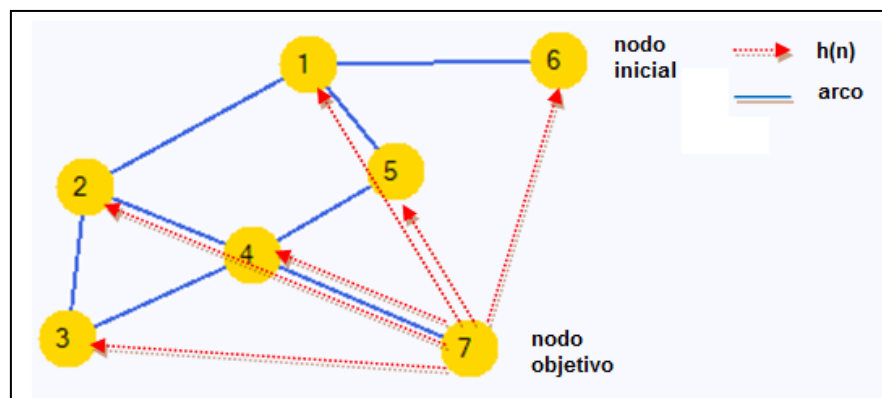


Figura 4.12: Distancia heurística calculada desde el nodo objetivo hacia todos los nodos
Fuente: Autores

El siguiente proceso empieza expandiendo los nodos desde el nodo inicial o nodo raíz, tomando como nodo padre al nodo actual en este caso nodo inicial y como nodos hijos a sus sucesores, donde cada sucesor contiene un coste $g(n)$ y la distancia en línea recta $h(n)$. La información de los nodos sucesores atraviesa un proceso de cálculo con el fin de elegir el camino más óptimo.

Este proceso se repite hasta encontrar un nodo objetivo siempre que este exista

Salida: La salida de este módulo es el camino de menor coste generado por el algoritmo de búsqueda y una matriz con el espacio de estados generado durante la búsqueda, ya sea en la simulación o haciendo uso del Robotino.

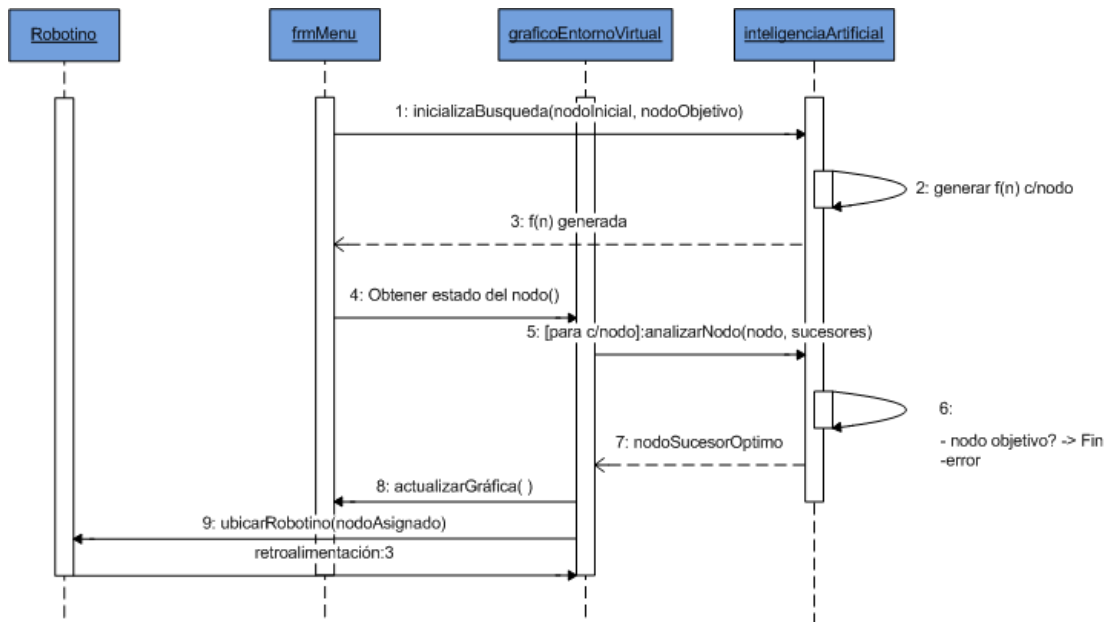


Figura 4.13: Diagrama de secuencia del Módulo Inteligencia Artificial
Fuente: Autores

Módulo Robotino

- **Entrada:** Este módulo tiene como entrada el estado de los sensores y los datos proporcionados por el módulo de IA.
- **Proceso:** Este módulo interactúa con todos los módulos que forman parte del sistema de búsqueda, enviando y recibiendo la información necesaria constantemente.

Las órdenes para el control del Robotino tales como seguidor inductivo, posicionamiento del nodo y detección de los caminos son generadas por este módulo con el fin de observar el funcionamiento del algoritmo de búsqueda en tiempo real. Las figuras 4.14 y 4.15 muestran algunos procesos realizados.

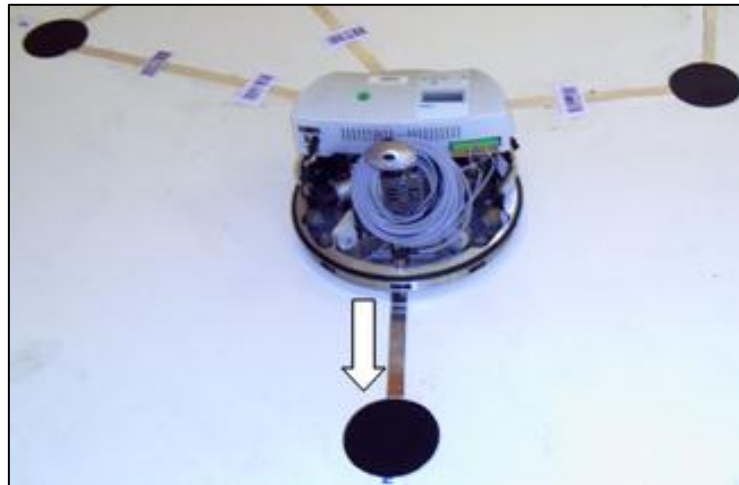


Figura 4.14: Robotino en el proceso de seguimiento del camino
Fuente: Autores

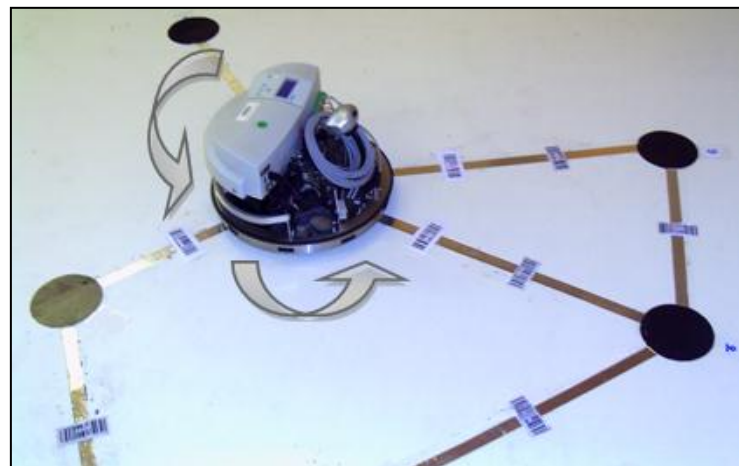


Figura 4.15: Robotino en el proceso de detección de caminos
Fuente: Autores

- **Salida:** La salida de este módulo son órdenes enviadas al sistema de control del Robotino, donde estas son ejecutadas mediante una secuencia de acciones controladas por el sistema de búsqueda.

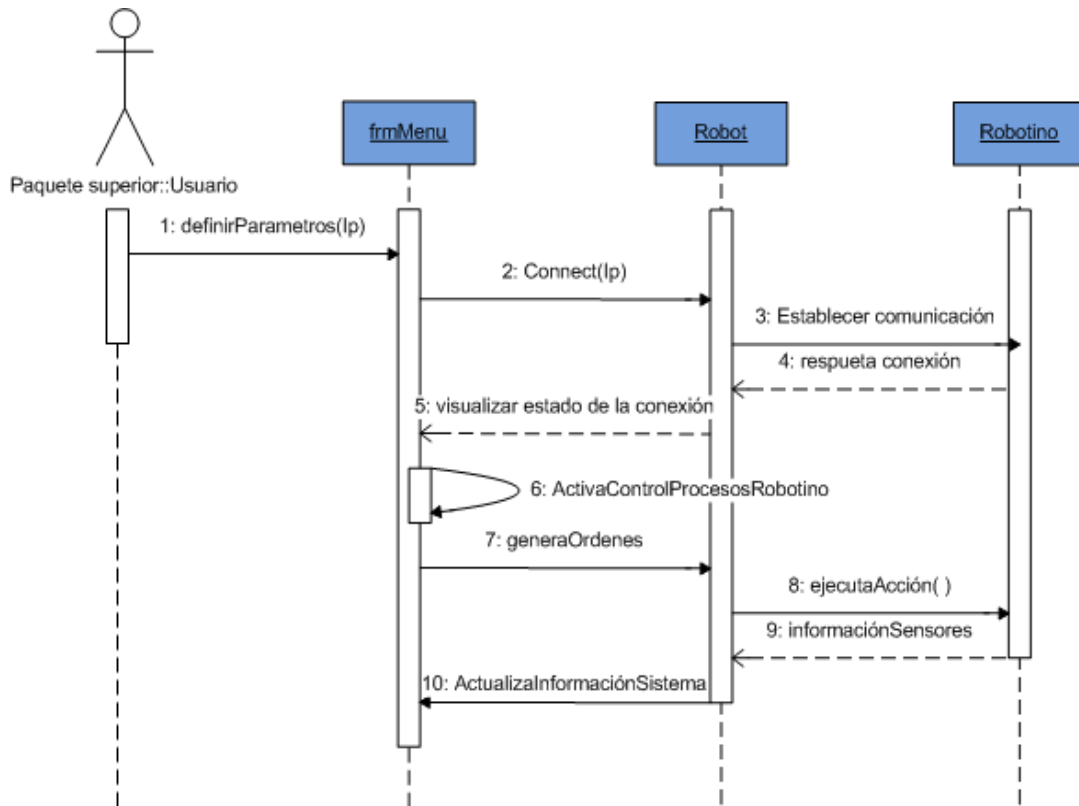


Figura 4.16: Diagrama de secuencia del Módulo Robotino

Fuente: Autores

4.3 HERRAMIENTAS DE SOFTWARE UTILIZADAS

Para el desarrollo de este proyecto de tesis se utilizaron las siguientes herramientas:

Librería BarcodeReader.dll

Biblioteca diseñada especialmente para el reconocimiento de códigos de barras en diferentes simbologías (código 128, código 39, interleaved 2 de 5, etc.). Soporta la lectura de varios formatos de imágenes (JPEG, GIF, BMP, TIFF, PNG), como también puede ser utilizada en lenguajes de programación como:

- ASP.NET, .NET Web Service

- C#, Vb.NET Windows Applications
- .NET, C#, VB.NET class library
- .NET Console Applications

Para el reconocimiento del código de barras, la librería emplea un procesamiento interno de la imagen, sin la necesidad de aplicar un previo procesamiento.

VintaSoftBarcode.Net SDK 5.2

Es una aplicación diseñada para la generación de códigos de barra. El usuario solo debe seleccionar el código de barras con el que desea trabajar, en nuestro caso seleccionamos Code128, ingresamos el valor del código, y por último guardar la imagen del código de barras generada por la aplicación. La imagen del código de barras se guarda con la extensión (.png).

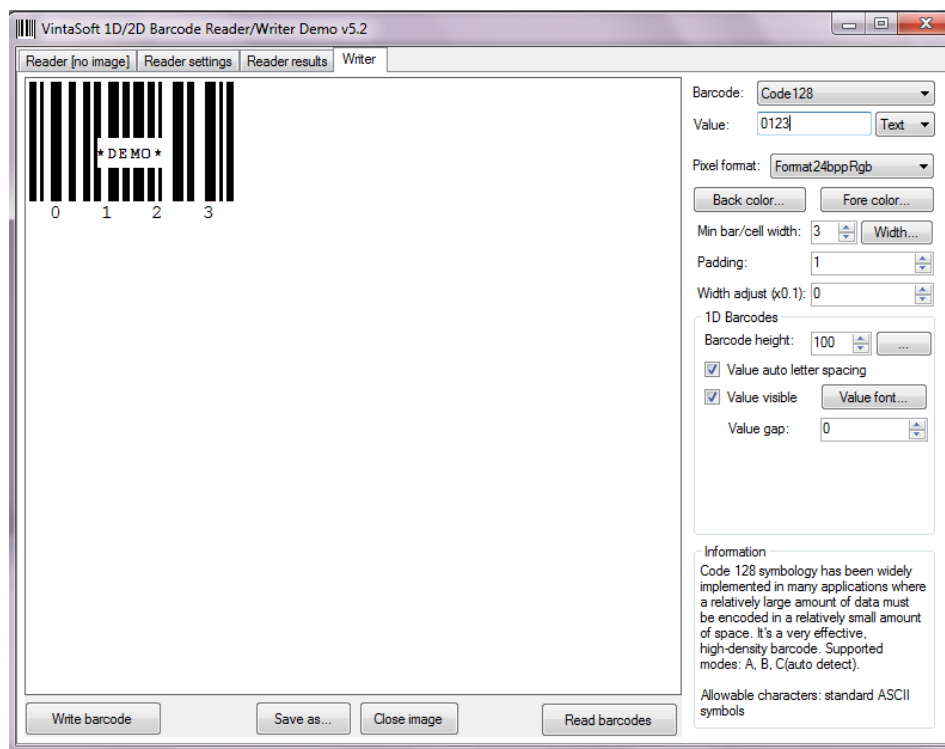


Figura 4.17: Generación del código de barras Code128

Fuente: Autores

Visual Studio C# .NET 2010

Es un lenguaje de programación orientado a objetos creado por Microsoft, utilizado para el rápido desarrollo de una gran variedad de aplicaciones con el poder C, C++, Visual Basic o Java.

Se adapta de manera natural al Framework y aprovecha al máximo todas sus características.

.Net Framework 4.0

Es el entorno de trabajo de la plataforma .NET., que gestiona la ejecución de programas escritos en C#, VisualBasic.Net, J#, C++ y Delphi. Todos estos lenguajes utilizan el mismo Framework para compilar las diferentes aplicaciones, por esta razón .NET Framework es considerado como la parte fundamental para el trabajo y ejecución de la tecnología .NET.

4.4 IMPLEMENTACIÓN DEL ALGORITMO

El sistema de búsqueda desarrollado utiliza el algoritmo A* para encontrar el camino más óptimo en un entorno de trabajo utilizando el Robotino de FESTO. Sin embargo, antes de ejecutar el proceso de búsqueda es necesario que el Robotino sea ubicado en el nodo inicial del entorno de trabajo, utilizando el panel de control del sistema de búsqueda. La figura 4.18 muestra el panel de control utilizado para la ubicación teleoperada del Robotino en el entorno de trabajo.

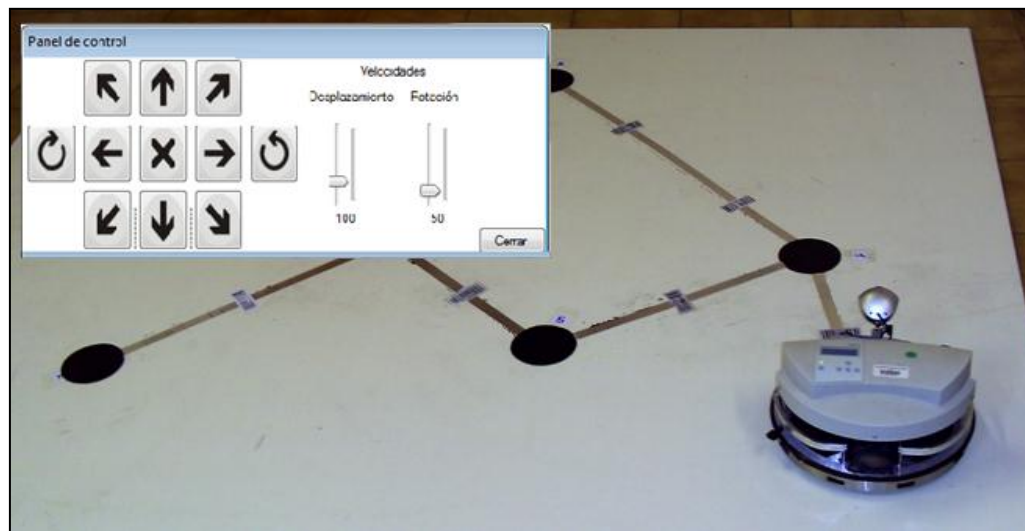


Figura 4.18: Ubicación del Robotino utilizando el panel de control
Fuente: Autores

Una vez realizado el proceso teleoperado por parte del usuario el robot móvil Robotino empieza su proceso en modo autónomo de la siguiente manera, el móvil inicia girando en sentido antihorario sobre el nodo detectando todos los posibles caminos. Cada camino contiene un código de barras que es capturado por la webcam del Robotino y a su vez este es enviado vía WLAN al sistema de búsqueda para el proceso de toma de decisiones. Cuando el código de barras del camino analizado concuerda con el camino elegido por el sistema de búsqueda, el Robotino gira hasta ubicarse en él camino y continúa con el proceso de desplazarse al siguiente nodo utilizando el sensor inductivo, el seguimiento termina cuando los sensores ópticos cambian de estado y el sensor inductivo deja de detectar el material metálico, ubicándose autónomamente sobre el nuevo nodo, el cual es analizado por el sistema de búsqueda con el fin de verificar si este se encuentra en el nodo objetivo y dar por terminado el proceso de búsqueda, de no ser así este proceso se repite con los siguientes nodos hasta llegar al nodo objetivo. En el caso de no identificar el código de barras del camino que lo lleve al siguiente nodo, el sistema de búsqueda genera otro camino óptimo que lo lleve desde su posición actual hacia el mismo objetivo.

Las figuras 4.19 y 4.21 muestran los caminos óptimos generados por el sistema utilizando el algoritmo de búsqueda A*, donde el color rojo representa el camino más óptimo, verde representa los nodos que fueron expandidos pero no analizados (nodos de la lista abiertos), celeste representa los nodos que fueron expandidos y analizados (nodos de la lista cerrados) y naranja son los nodos del grafo que no han sido expandidos ni analizados, además al lado derecho de la ventana se puede apreciar los resultados obtenidos del proceso de búsqueda.

En cuanto a las figuras 4.20 y 4.22 muestran los caminos seguidos por el Robotino haciendo uso del sistema de búsqueda para desplazarse desde uno nodo inicial a un nodo objetivo.

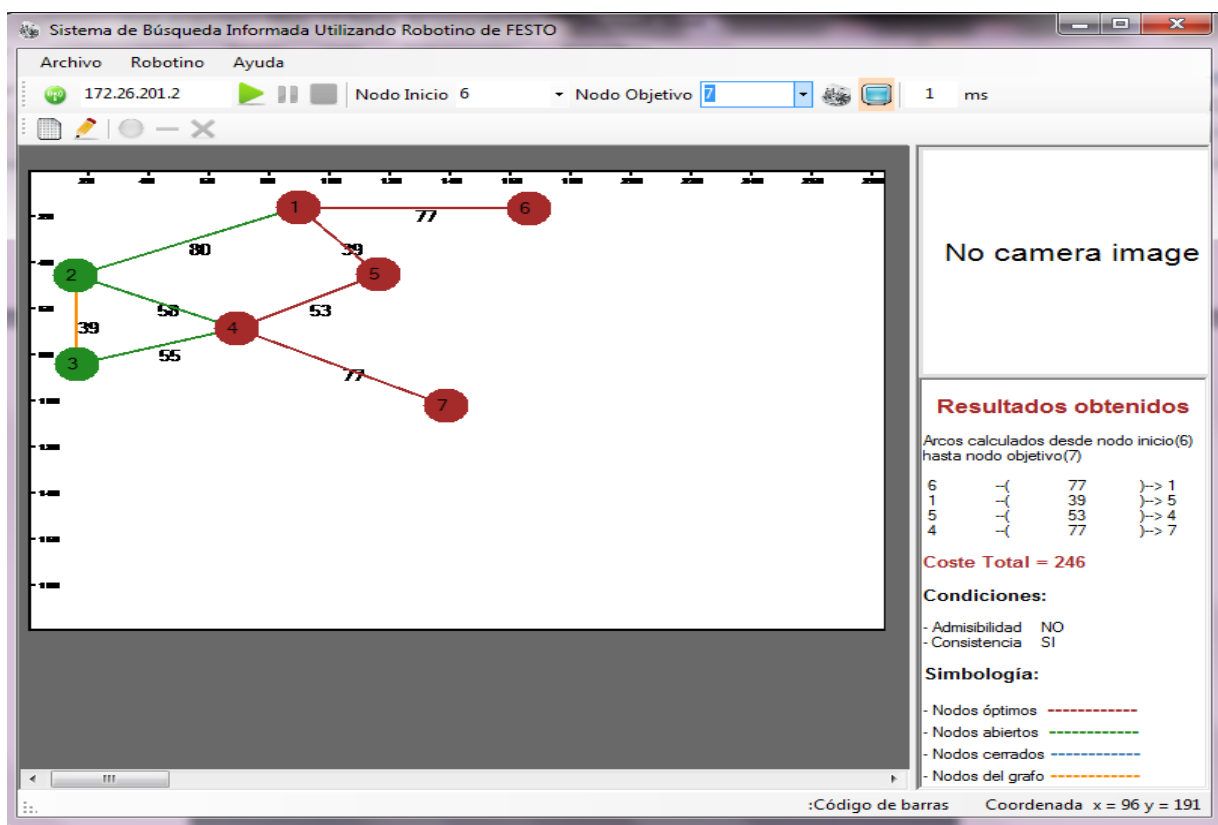


Figura 4.19: Camino encontrado por el simulador para ir del nodo 6 al nodo 7

Fuente: Autores



Figura 4.20: Camino seguido por el Robotino para ir del nodo 6 al nodo 7
Fuente: Autores

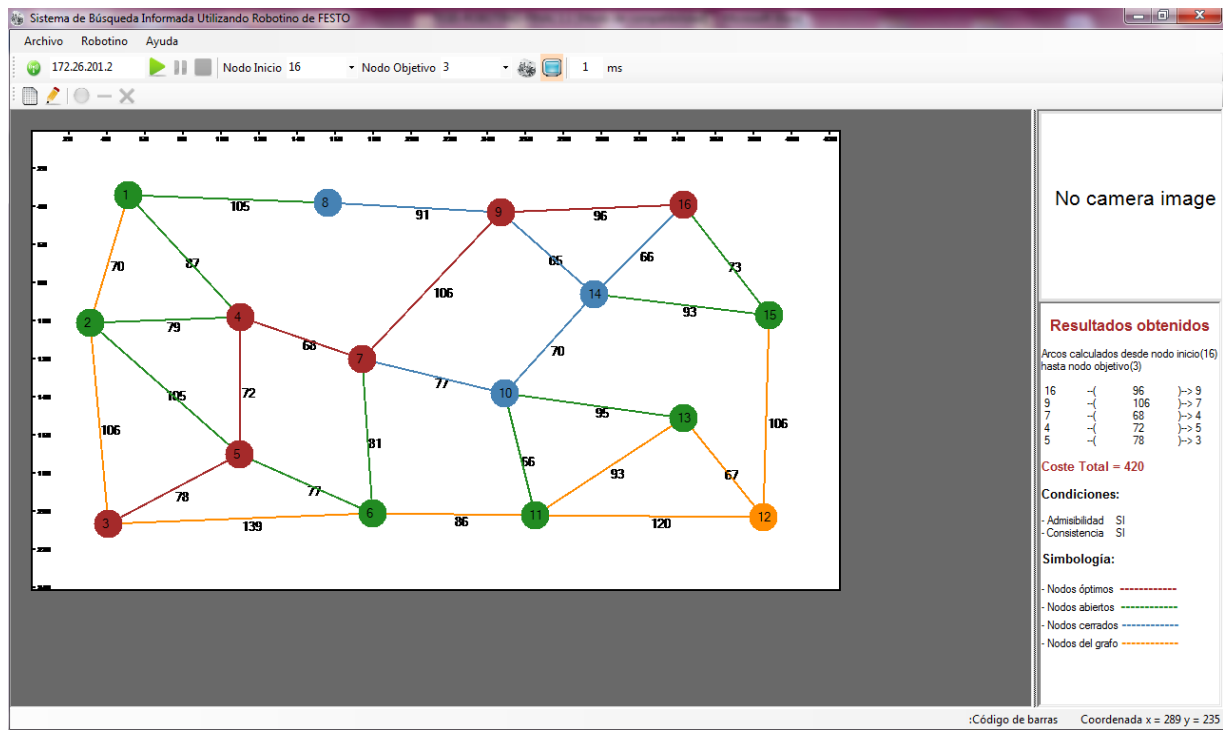


Figura 4.21: Camino encontrado por el simulador para ir del nodo 16 al nodo 3
Fuente: Autores

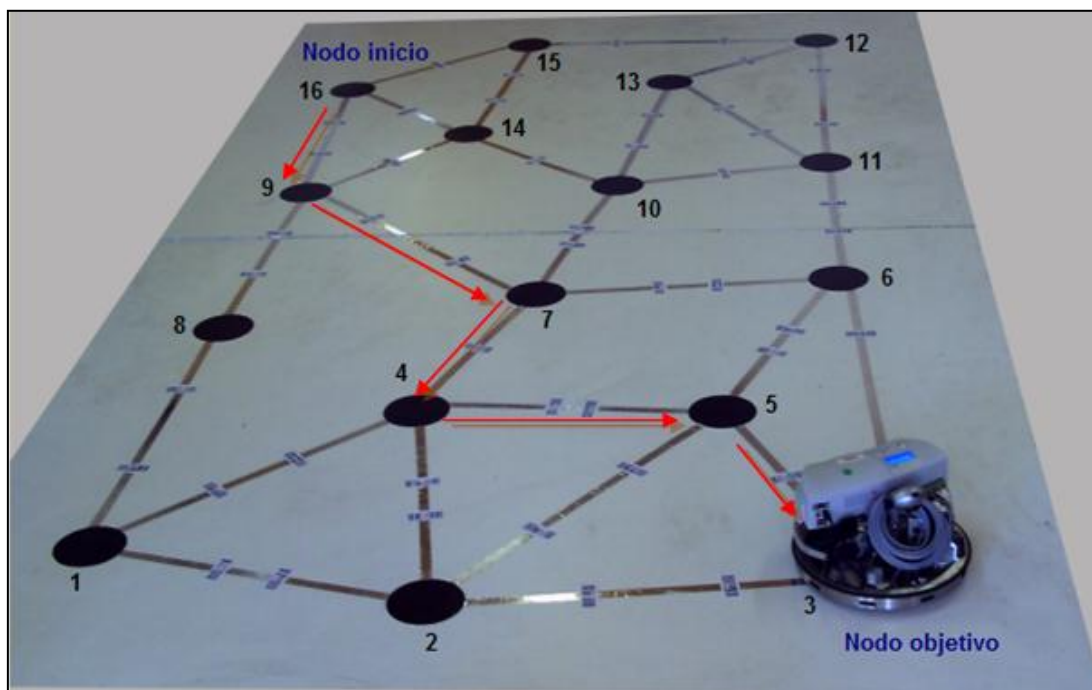


Figura 4.22: Camino seguido por el Robotino para ir del nodo 16 al nodo 3
Fuente: Autores

4.5 PRUEBAS REALIZADAS

Las pruebas se pueden dividir en dos partes: pruebas de software y la más importante las pruebas con el Robotino, es decir como interactúa el robot móvil con el software, ya que solo de esa manera se puede comprobar el funcionamiento del sistema de búsqueda en tiempo real.

4.5.1 PRUEBAS DE SOFTWARE

En la tabla 4.2 se muestra las pruebas realizadas con algunas herramientas de software, siendo estas necesarias para el buen funcionamiento del sistema de búsqueda utilizando el Robotino en tiempo real.

#	Descripción	Resultado esperado	Observación	Efectividad %
1	Comunicación entre el Robotino		No se logró	50%

	y la Pc usando el API para C#.NET bajo Windows Vista	Comunicación bidireccional	obtener respuesta del estado de los sensores del Robotino	
2	Comunicación entre el Robotino y la Pc usando el API para C#.NET bajo Windows XP y Win7		Se consiguió enviar y recibir datos satisfactoriamente	100%
3	Búsqueda realizada en grafos y árboles creados en el entorno virtual	Encontrar el camino más óptimo	El sistema de búsqueda consigue resolver el problema eficientemente	100%
4	Librería VintaSoftBarcode.Net y Librería Free BarcodeReader	Valor del código de barras	Se requería instalación y presentaba varias limitaciones	30%
5	Librería BarcodeLib.BarcodeReader		No requiere instalación, su funcionamiento presenta pocas limitaciones que pueden ser corregidas aplicando programación	90%

Tabla 4.2: Pruebas de Software

Fuente: Autores

4.5.2 PRUEBAS SOBRE EL ROBOTINO

En la tabla se muestran las pruebas realizadas para el control del Robotino con el fin de observar el funcionamiento del sistema de búsqueda en tiempo real.

#	Descripción	Resultado esperado	Observación	Efectividad %
1	Seguimiento de un camino utilizando sensores ópticos		Problemas al colocar el código de barras sobre el camino	60%
2	Seguimiento de un camino utilizando un sensor inductivo		La división de voltaje producida	20%

	con dos materiales metálicos (aluminio y latón)	Desplazamiento de un nodo a otro	en los extremos de los materiales ocasionaba que Robotino se desvié del camino	
3	Seguimiento de un camino utilizando un sensor inductivo con un solo material metálico (aluminio) definiendo un límite interior y un límite exterior.		Robotino logra atravesar el camino exitosamente. Sin embargo, la capacidad del procesamiento del computador causa problemas en el tiempo de respuesta a través de una WLAN, produciendo salidas del camino en algunas ocasiones.	85%
4	Posicionamiento del Robotino sobre el nodo utilizando los encoders	Distancia desplazada	Se obtuvo buenos resultados, aunque la distancia varía de acuerdo a la velocidad que se manipule en los motores	90%

Tabla 4.3: Pruebas sobre el Robotino

Fuente: Autores

Se optó por utilizar un solo sensor inductivo por ser más eficientes que los sensores ópticos ante las variaciones del ambiente (exceso de luz) y posibles residuos (polvo, agua, papel, etc.). Esto finalmente permitió obtener buenos resultados aunque no tan eficiente como se lo haría utilizando dos sensores inductivos.

Las pruebas realizadas sobre el Robotino nos permitieron definir una velocidad mínima, máxima y óptima con el fin de llevar al robot móvil a su destino de forma

eficiente. En el proceso de búsqueda el Robotino, normalmente se ejecuta con una velocidad constante para que la dinámica del robot móvil y el tiempo de respuesta entre la Pc y el Robotino no afecten el seguimiento del camino. La tabla 4.4 muestra los valores definidos para cada velocidad y la distancia recorrida en cm/min calculada utilizando la siguiente fórmula:

$$D=2*3,14*40/10=25,12 \text{ cm por c/d vuelta}$$

$$D=(25,12/100)*80 = 20,09 \text{ m/min}$$

$$D= 20,09 \text{ m/min}$$

Dónde:

Radio de la rueda=40mm

RPM=80

Velocidad	Tarea	Rpm	Distancia recorrida (m/min)
Mínima	Seguidor avance	m1:-80 m2:0 m3:80	20
	Seguidor giro (izq. ó der.)	m1:-80 m2:0 m3:80	20
	Barrido imagen	m1:50 m2:50 m3:50	12,5
	Giro entre caminos	m1:80 m2:80 m3:80	20
	Giro posición metal	m1:-80 m2:-80 m3:-80	20
	Centrado	m1:-80 m2:0 m3:80	20
	Seguidor avance	m1:-400 m2:0 m3:400	100,4
	Seguidor giro (izq. ó der.)	m1:-375 m2:50 m3:500	109,8

Óptima	Barrido imagen	m1:80 m2:80 m3:80	20
	Giro entre caminos	m1:300 m2:300 m3:300	75,36
	Giro posición metal	m1:-100 m2:-150 m3:-100	25,12
	Centrado	m1:-400 m2:-0 m3:400	100,48
Máxima	Seguidor avance	m1:600: m2:-0 m3:-600	150,72
	Seguidor giro (izq. ó der.)	m1:-600 m2:100 m3:600	150,72
	Barrido imagen	m1:150 m2:150 m3:150	37,6
	Giro entre caminos	m1:600 m2:600 m3:600	150,7
	Giro posición metal	m1:-600 m2:-600 m3:-600	150,7
	Centrado	m1:-600 m2:0 m3:600	150,7

Tabla 4.4: Velocidades del Robotino
Fuente: Autores

4.6 CONDICIONES DEL ENTORNO DE TRABAJO

Para ejecutar el sistema de búsqueda utilizando el Robotino hay que tomar en cuenta ciertas condiciones en la construcción del entorno de trabajo como:

- El grafo diseñado en el entorno virtual deber ser construido sobre una superficie plana de preferencia de color blanco para conseguir óptimos resultados.

- Los caminos no deben ser inferiores a 65cm de longitud debido al tamaño del Robotino, ya que los códigos de barras de cada camino podrían quedar bajo su chasis, ocasionando problemas al momento de capturar la imagen de código de barras.

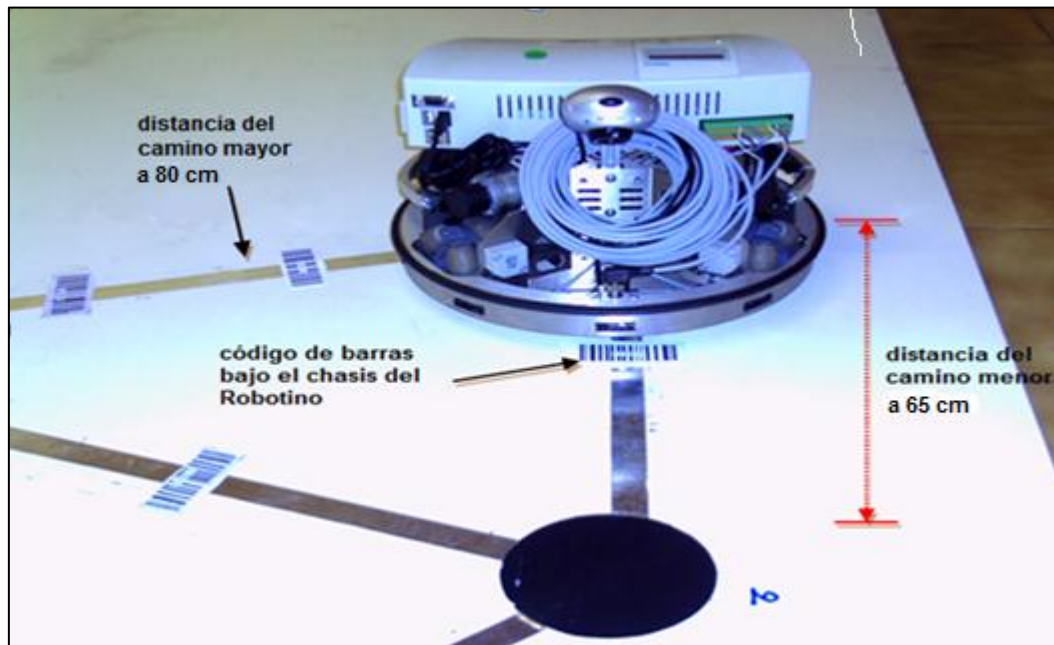


Figura 4.23: Robotino en un camino menor a 65 cm
Fuente: Autores

- La iluminación del entorno de trabajo debe ser controlada con el fin de evitar variaciones en la lectura de los sensores ópticos y en las imágenes capturadas por la WebCam del Robotino.

Las velocidades predefinidas en el sistema de búsqueda para el desplazamiento e identificación de caminos pueden ser variadas por el usuario. Sin embargo, debido al tiempo de respuesta entre la Pc y el Robotino no se pueden obtener buenos resultados a grandes velocidades.

Una vez tomadas en cuenta las condiciones descritas anteriormente el sistema requiere la interacción del usuario para generar el nodo inicial, el nodo objetivo, posicionamiento teleoperado del Robotino y poner en funcionamiento al algoritmo de

búsqueda, una vez iniciado el algoritmo de búsqueda el Robotino junto con el programa informático actúa de manera autónoma hasta situar al robot móvil en el nodo objetivo.

4.7 RESULTADOS OBTENIDOS

Una vez solucionados los inconvenientes tanto de hardware como de software y el ambiente del entorno de trabajo, se puede decir que los objetivos planteados al inicio de este proyecto de tesis fueron conseguidos satisfactoriamente.

Con un robot móvil ya desarrollado como es Robotino de FESTO se ha logrado tener una mayor precisión en sus movimientos. Esto permite que el robot móvil realice eficientemente las tareas ordenadas por el sistema de búsqueda para atravesar el camino más óptimo generado por el algoritmo de búsqueda informada A*.

Las librerías elegidas para el tratamiento de imágenes y control total del Robotino han permitido ahorrar tiempo, ofreciéndonos un conjunto de funciones necesarias para la comunicación bidireccional entre la Pc y el Robotino, con el fin de enviar y recibir los datos necesarios para la eficiente ejecución del sistema de búsqueda utilizando el Robotino.

En cuanto al funcionamiento del algoritmo de búsqueda informada implementado en el Robotino se obtuvo buenos resultados. En un principio se trató que el Robotino ejecute todo el proceso de búsqueda. Esto requería mayor tiempo en la solución del problema, razón por la cual se optó por optimizar el tiempo haciendo que el Robotino se desplace por el camino más óptimo generado por el algoritmo de búsqueda A*.

El tiempo de respuesta en el envío y recepción de la información necesaria a través una WLAN fue uno de los inconvenientes más cruciales, lo que nos llevó a utilizar un computador con mayores capacidades de procesamiento, a una dual core, logrando

mejorar significativamente este inconveniente en la percepción y ejecución de acciones utilizando el Robotino. Este retardo en la transmisión de la información ocasionaba que el Robotino se desvié del camino, lo que nos llevó a aumentar el ancho de la cinta metálica, solucionando de esta manera el inconveniente y consiguiendo buenos resultados en caminos sin curvas.

Finalmente, otro de los inconvenientes fue la desconexión automática que se realizaba internamente en el Robotino durante el seguimiento del camino más óptimo generado por el algoritmo de búsqueda informada A^* , ocasionando que el robot móvil no logre cumplir todas las tareas. Este inconveniente se evitó verificando el estado de la conexión y reconectándolo automáticamente cuando sea necesario durante la ejecución del sistema de búsqueda.

CONCLUSIONES Y RECOMENDACIONES

En este apartado se citan algunas conclusiones y recomendaciones a las que se llegaron dando durante el desarrollo del presente proyecto de titulación, las cuales son de vital importancia para aquellas personas que estén interesadas en profundizar su conocimiento en campos como la robótica móvil y la Inteligencia Artificial.

CONCLUSIONES

1. Se obtuvo un sistema inteligente que implementa un algoritmo de búsqueda informada A* para encontrar el camino más óptimo (camino con el coste más barato) utilizando el robot móvil Robotino de FESTO, realizando percepciones y acciones en tiempo real de cada tarea encomendada, con el fin de desplazarse por el camino más óptimo generado por el algoritmo de búsqueda.
2. El API utilizado para el control del Robotino es una de las herramientas que nos permite trabajar con lenguajes de alto nivel, realizar aplicaciones que requieren grandes capacidades computacionales con mayor flexibilidad, sin la necesidad de estar limitado a un software basado en diagrama de bloques como es el Robotino View, explotando de esta manera todo el potencial del Robotino en las aplicaciones que se consideren necesarias.
3. El algoritmo implementado en el sistema de búsqueda como es el A*, funciona eficientemente encontrando el camino más óptimo siempre que este exista, en el sentido de que aprovecha de mejor manera la memoria del sistema, permitiendo a las aplicaciones ser más rápidas, eficientes y robustas, siendo empleadas en aplicaciones como: hallar el camino más corto entre dos ciudades es algo que interesa a muchos turistas, en la aeronavegación, el pilotaje automático busca las rutas más próximas, en los video juegos de estrategia, busca el camino más corto (Warcraft, Pacman), etc.

4. Se optó por utilizar un sensor inductivo para el seguimiento y detección de caminos por ser inmune a factores como la luz, humedad, temperatura, impurezas, etc., proporcionando al sistema de búsqueda datos eficientes en sus lecturas para la toma de decisiones, tomando en cuenta que este sensor debe estar a una altura máxima de 2mm del piso para conseguir óptimos resultado en su funcionamiento.
5. El tiempo y espacio requeridos para la solución del problema utilizando el algoritmo de búsqueda A*, dependen mucho de la cantidad de nodos y arcos que contenga el entono virtual diseñado por el usuario.
6. La presentación de resultados en forma gráfica y detallada de cada paso realizado por el algoritmo de búsqueda, facilitan la comprensión del funcionamiento del algoritmo, con el fin de orientar de mejor manera a las personas interesadas en el tema.
7. La configuración omnidireccional del robot móvil Robotino de FESTO facilitó la ejecución de movimientos en cualquier dirección, por ejemplo, girar 360° sobre su propio eje para la identificación de caminos.

RECOMENDACIONES

1. Para el funcionamiento del sistema de búsqueda por primera vez se recomienda instalar el API para Visual Studio .NET 2010 que permite establecer la comunicación entre la Pc y el Robotino de FESTO. El API está disponible en la siguiente dirección:
<http://forum.openrobotino.org/showthread.php?10-Current-C-C-Java-.Net-API>

2. Cuando el sistema de búsqueda hace uso del Robotino de FESTO para encontrar el camino más óptimo es necesario que en los extremos de cada camino del entorno de trabajo se tenga un código de barras etiquetado, el valor de los mismos también deben ser definidos en el sistema que lo controla permitiendo que de esta manera el sistema tome las decisiones y envíe las órdenes necesarias para la ejecución de acciones sobre el Robotino de FESTO. Hay que tomar en cuenta también que el primer dígito de cada código de barras generado por cualquier programa informático es ignorado por el sistema de búsqueda, razón por la cual se recomienda tomar el valor desde su segundo dígito.
3. Para un óptimo funcionamiento del sistema de búsqueda utilizando el Robotino de FESTO se recomienda utilizar las velocidades establecidas en la tabla 4.4, con el fin de evitar que el robot móvil se desvíe del camino durante el proceso del seguimiento, y logre realizar una identificación eficiente de los códigos de barra situados en cada camino utilizando la WebCam, de no ser así, esto ocasionaría inconvenientes en las acciones realizadas por el robot móvil que a su vez son ordenadas por el sistema que lo controla.
4. Para trabajos a futuro utilizando lenguajes de alto nivel a través de una comunicación WLAN con el Robotino de FESTO, se pueden realizar mejoras en el tiempo de respuesta (envío y recepción de datos) para que el robot móvil realice las acciones con mayor rapidez, por lo que sería de gran ayuda investigar la manera de cargar en el robot móvil una aplicación desarrollada bajo cualquiera de los lenguajes soportados por él.
5. Otro aspecto fundamental que se puede mejorar es la desconexión automática que se produce en el Robotino de FESTO durante la ejecución de tareas, ocasionando inconvenientes en el cumplimiento de las mismas

REFERENCIAS BIBLIOGRÁFICAS

- **[Nilson Nils 2000]** NILSON, Nils J., *Inteligencia Artificial Una Nueva Síntesis*, 1^{ra}. Edición, Editorial McGraw-Hill, Madrid-España, 2000.
- **[Ollero Aníbal 2001]** OLLERO BARTURONE, Aníbal, *ROBÓTICA – Manipuladores y robots móviles*, 1^{ra}. Edición, Editorial Marcombo, Barcelona-España, 2001.
- **[Pajares Gonzalo 2006]** PAJARES MARTINSANZ, Gonzalo y SANTOS PEÑAS, Matilde, *Inteligencia Artificial e Ingeniería del Conocimiento*, 1^{ra}. Edición, Editorial Alfaomega, México, 2006.
- **[Pallás Ramón, 2005]** PALLÁS ARENY, Ramón, *Sensores y Acondicionadores de Señal*, 4^{ta}. Edición, Editorial Marcombo, Barcelona-España, 2005.
- **[Rich Elaine 1994]** RICH, Elaine y KNIGHT, Kevin, *Inteligencia Artificial*, 2^{da}. Edición, Editorial McGraw-Hill, Madrid-España, 1994.
- **[Russell Stuart 2004]** RUSSELL, Stuart J. y NORVING Peter, *Inteligencia Artificial Un Enfoque Moderno*, 2^{da}. Edición, Editorial Pearson Educación S.A., Madrid-España, 2004.
- **[Siegwart Roland 2004]** SIEGWART, Roland y NOURBAKHSH, Illah Reza, *Introduction to Autonomous Mobile Robots*, 1^{ra}. Edición, Editorial MIT Press., London-England, 2004.

PÁGINAS WEB

- **[Actuadores]** ACTUADORES

Esta página muestra la clasificación de los actuadores, que permiten el movimiento de los elementos de un robot.

<http://proton.ucting.udg.mx/materias/robotica/r166/r68/r68.htm>

➤ **[Festo Didactic] FESTO DIDACTIC**

Página oficial de FESTO donde se encuentra información acerca de todos sus productos.

<http://www.festo-didactic.com>

➤ **[Locomoción por orugas] LOCOMOCION POR ORUGAS**

Página oficial de la empresa iRobot donde se puede apreciar varios robots con locomoción por orugas.

http://www.irobot.com/gi/ground/710_warrior/

➤ **[Luchadores de sumo Humanoides] LUCHADORES DE SUMO HUMANOIDES**

Muestra las pruebas que se desarrollaron en el concurso SonarMática llevado a cabo en la Universidad de Cataluña en el 2010.

http://2010.sonar.es/es/sonarmatica/aess-iri-humanoid-lab_35.html

➤ **[Robocup] ROBOCUP**

Esta página muestra las diferentes categorías de competición en la Robocup del 2010.

http://www.roboticspot.com/especial/robocup2010/robocup_2010_2.php

➤ **[Robot Bípedo] ROBOT BIPEDO**

Página oficial de Asimo que presenta la historia, características, galería de imágenes, etc.

<http://asimo.honda.com/>

➤ **[Robot con ruedas] OROBOT CON RUEDAS**

Esta página muestra algunos de los robots creados por la NASA

<http://www.ovaliente.com.ar/perso/robot/nasa.htm>

➤ **[Robot cuadrúpedo] ROBOT CUADRÚPEDO**

Página oficial del robot Aibo creado por la empresa Sony

<http://support.sony-europe.com/aibo/>

➤ **[Robot de asistencia] ROBOT DE ASISTENCIA**

En esta página se encuentra el robot de asistencia Japonés Yurina creado por la empresa Toyota, para el cuidado de pacientes.

<http://www.discapacidadonline.com/yurina-robot-asistencial-japones.html>

➤ **[Robot de limpieza de piscinas] ROBOT DE LIMPIEZA DE PISCINAS**

En esta página se puede apreciar un robot construido para la limpieza de piscinas creado por Eco Pool Technologies.

<http://www.solar-breeze.com/>

➤ **[Robot de limpieza del hogar] ROBOT DE LIMPIEZA DEL HOGAR**

Características técnicas del robot automático Roomba 531

<http://www.appinformatica.com/domotica-irobot-roomba-531-robot-de-limpieza-automatiko.php>

➤ **[Robot de limpieza industrial] ROBOT DE LIMPIEZA INDUSTRIAL**

En esta página se puede ver un robot diseñado para la limpieza en industrias construido por la empresa Trenz-Cleaneer.

http://www.trenzcleaner.com/trenz_cleaner.htm

➤ **[Robot de vigilancia] ROBOT DE VIGILANCIA**

Robot mSecurit de la empresa MoviRobotics

<http://www.movirobotics.com/SPmsecurit.php>

- **[Robot hexápodo – Athlete Rover]** ROBOT HEXÁPODO - ATHLETE ROVER
<http://www-robotics.jpl.nasa.gov/systems/system.cfm?System=11>
- **[Robot hexápodo – Silo6]** ROBOT HEXÁPODO
Descripción del robot Silo6
<http://www.iai.csic.es/users/silo6/>
- **[Robot militar]** ROBOT MILITAR
Robot militar R-Gator con kernel GNU/Linux de la empresa IRobot desarrollado para aplicaciones militares.
<http://GNU/Linuxdom.wordpress.com/2009/09/08/robot-militar-con-kernel-GNU/Linux-r-gator-xd/>
- **[Robot rastreador]** ROBOT RASTREADOR
Esta página muestra el funcionamiento de un robot rastreador y sus principales componentes.
<http://www.neoteo.com/robot-siguelineas-teoria-y-practica.neo>
- **[Robots de competencia]** ROBOTS DE SUMO
La página muestra imágenes y resultados de las pruebas de sumo llevadas a cabo en el museo de Ciencia de Boston.
<http://www.machinescience.org/gallery/robot-sumo-2010/>
- **[Robots móviles]** ROBOTS MÓVILES
Esta página muestra las configuraciones típicamente utilizadas en la robótica móvil.
<http://www.muchotrasto.com/TiposDePlataformas.php>

➤ **[Rovio] ROBOT DE TELEVIGILANCIA Y TELEPRESENCIA**

Esta página muestra las características técnicas del robot Rovio.

<http://www.robotica-personal.es/2009/02/rovio-es-un-robot-wi-fi-con-una-webcam.html>

➤ **[Sensores internos] SENSORES INTERNOS**

La página presenta una breve introducción a los sensores internos que permiten obtener en la mayoría de los robots la posición, velocidad y aceleración.

<http://www.dccia.ua.es/dccia/inf/asignaturas/ROB/optativos/Sensores/internos.htm>
!

➤ **[Sensores Proximidad] SENSORES DE PROXIMIDAD**

Tipos de sensores de proximidad según el principio físico que utilizan.

<http://sensoresdeproximidad.galeon.com/>

➤ **[Sensores] SENSORES**

En esta página se puede encontrar conceptos generales de los sensores.

http://robots-argentina.com.ar/Sensores_general.htm

➤ **[Vehículos de guiado automático] VEHÍCULOS DE GUIADO AUTOMÁTICO**

Esta página muestra los diferentes vehículos AGV utilizados en la industria para el transporte de objetos de diferente tamaño y peso.

http://www.coreconagvs.com/products/E210_whitepaper.php

MANUALES TÉCNICOS

➤ Manual Robotino, Festo, 2007

ARCHIVOS DE ACROBAT READER (.PDF)

- **[Arquitectura de Control para Robots]** JAMSHIDI, M., *Fuzzy Tracking Methos for Mobile Robots*, Dto. de Ingeniería de Sistemas y Automática, Universidad de Sevilla.
- **[Cazorla Miguel]** CAZORLA, Miguel, *Robots Autónomos*, Dto. Ciencia de la Computación e I.A., Universidad Alicante.
- **[Encoder Absoluto]** ELTRA, *Encoder Absoluto descripción general*, 2000.
- **[García Juan 2007]** GARCÍA SÁNCHEZ, Juan Enrique, *Sensores*, Dto. De Ing. Electrónica y Automática, Universidad de Castilla, Noviembre 2007.
- **[Muñoz V.]** MUÑOZ MARTÍNES, V., y otros, *Modelado Cinemático y Dinámico de un Robot móvil Omni-Direccional*, Dto. Ingeniería de Sistemas y Automática, Universidad de Málaga.
- **[Sensores Inductivos]** CANTO Q., Carlos E., Facultad de Ciencias/UASLP.
- **[Sucar Enrique]** SUCAR, L. Enrique, *Introducción a la Robótica*, Instituto Nacional de Astrofísica, Óptica y Electrónica.
- **[Tipos de ruedas]** ORTIGOZA, R. Silvia y otros, *Una panorámica de los Robots Móviles*, Revista Electrónica de Estudios Telemáticos, Universidad Rafael Belloso Chacín, 2007, p. 5.

TESIS

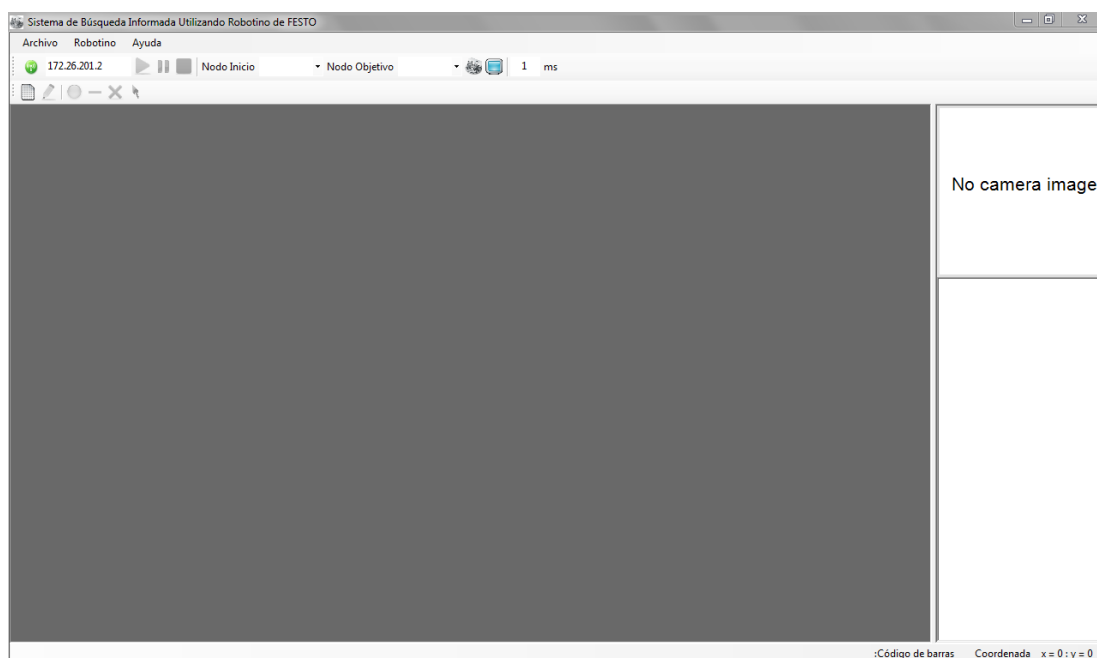
- **[Robots de competencia]** JIMÉNEZ GAÑÁN, Luis, *Robot Velocista de Competición basado en Corba*, marzo de 2006.

ANEXO 1

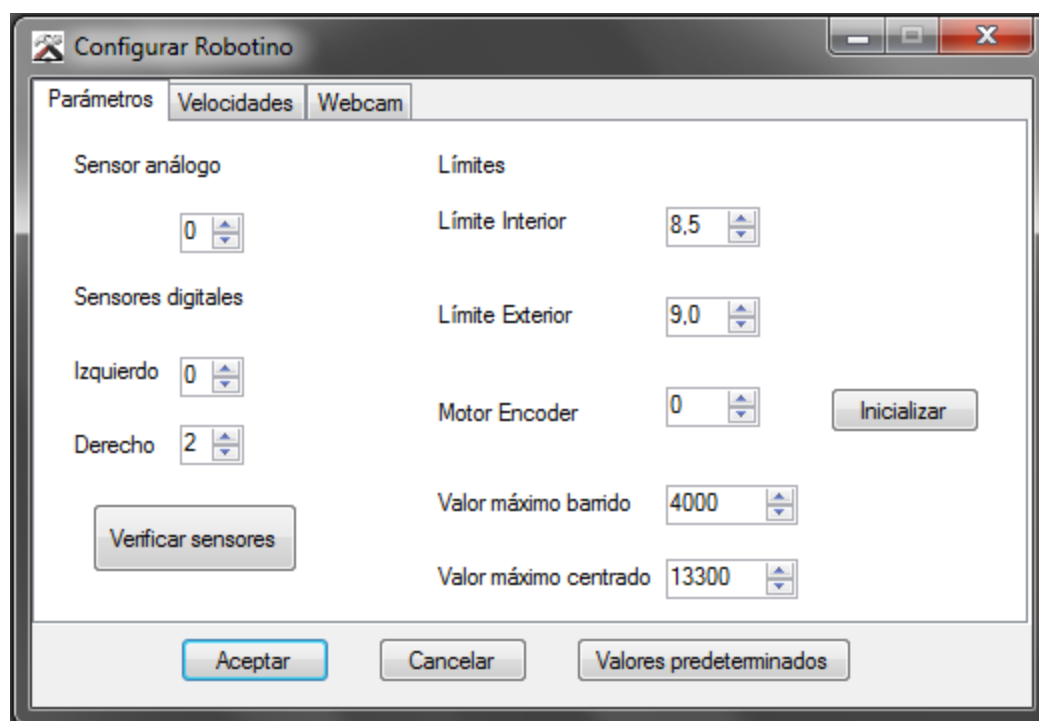
Pantallas del Sistema de Búsqueda Informada Utilizando

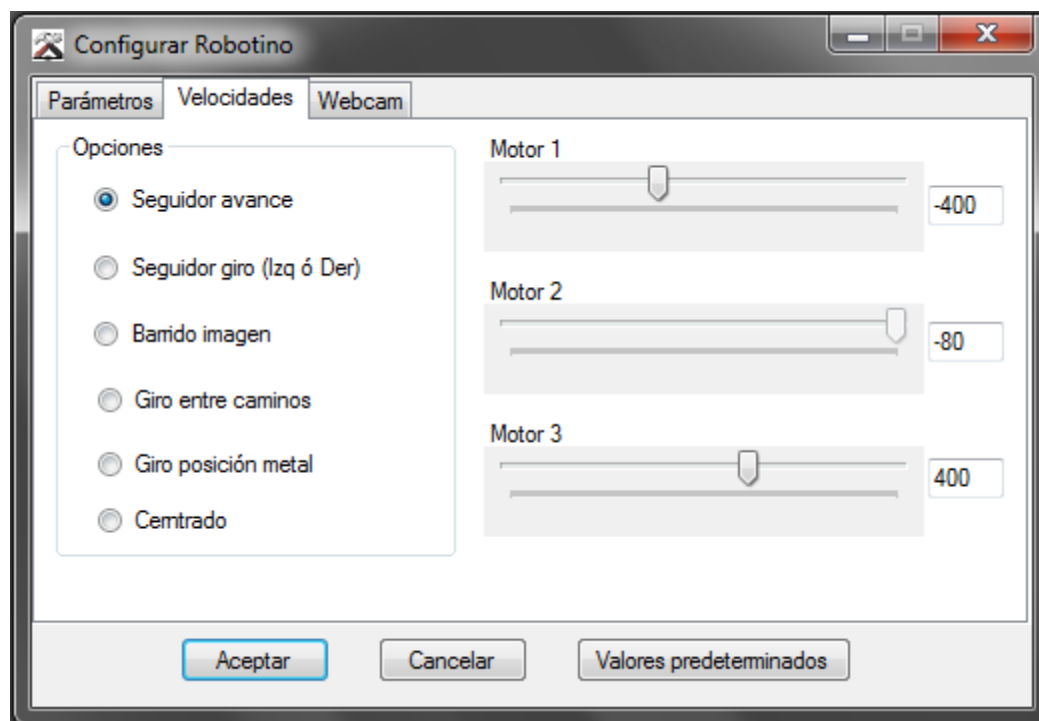
Robotino de FESTO

Pantalla principal del Sistema

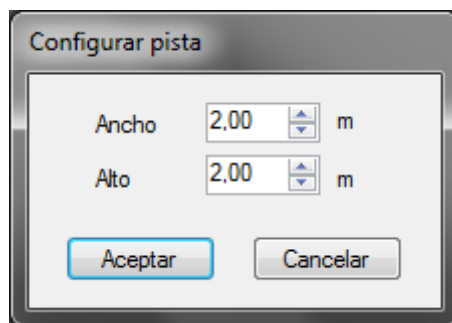


Pantallas de configuración del Robotino

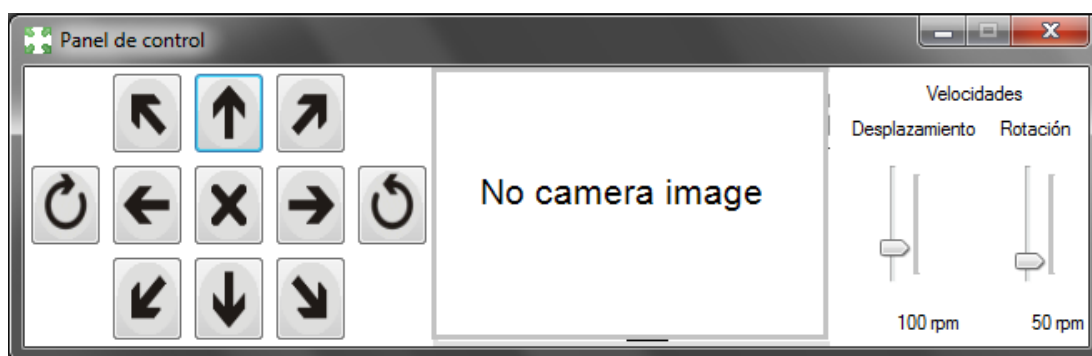




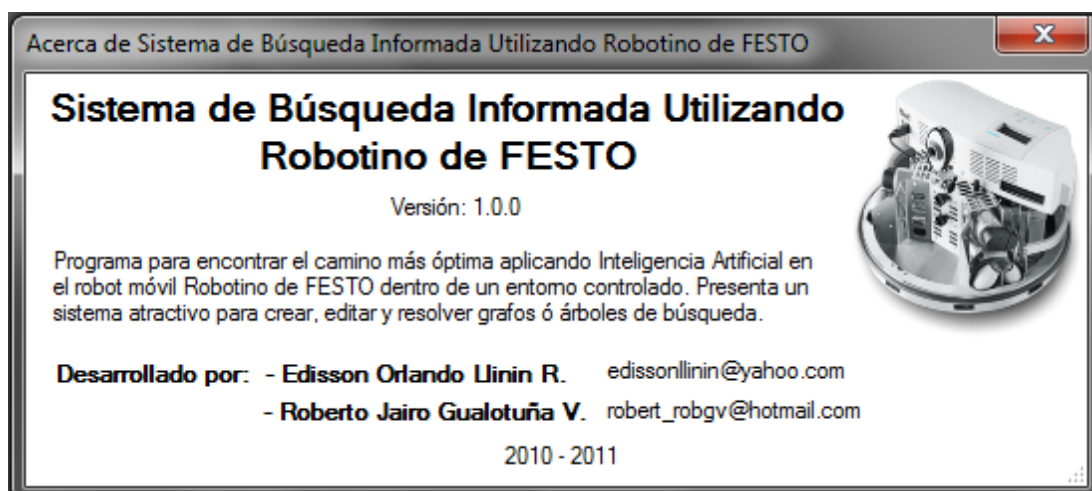
Pantalla para configuración del Entorno Virtual



Pantalla para el control teleoperado del Robotino



Pantalla Acerca de



ANEXO 2
Manual de usuario

SISTEMA DE BÚSQUEDA INFORMADA UTILIZANDO ROBOTINO DE FESTO MANUAL DE USUARIO

SISTEMA DE BÚSQUEDA INFORMADA UTILIZANDO ROBOTINO DE FESTO

Este producto está orientada para aquellas personas interesadas en el control del robot móvil Robotino de FESTO y la utilización de algoritmos de búsqueda informada.



REQUISITOS DEL SISTEMA

Asegúrese de que el sistema informático de la PC cumpla con los siguientes requisitos:

- PC con Windows XP/Seven
- 2Gb (o más) RAM
- 2Ghz (o más) procesador
- WLAN (Red de Área Local Inalámbrica)
- Microsoft Visual Studio 2010
- API para Robotino
- Robot móvil Robotino de FESTO

El API se puede descargar de la siguiente dirección:
<http://wiki.openrobotino.org/index.php?title=Downloads>

Nota: Para óptimos resultados se recomienda utilizar computadores superiores a una Dual Core.

➤ INSTALACIÓN DEL API

Una vez descargado el API desde el sitio web, ejecutarlo y escoger la opción en inglés.

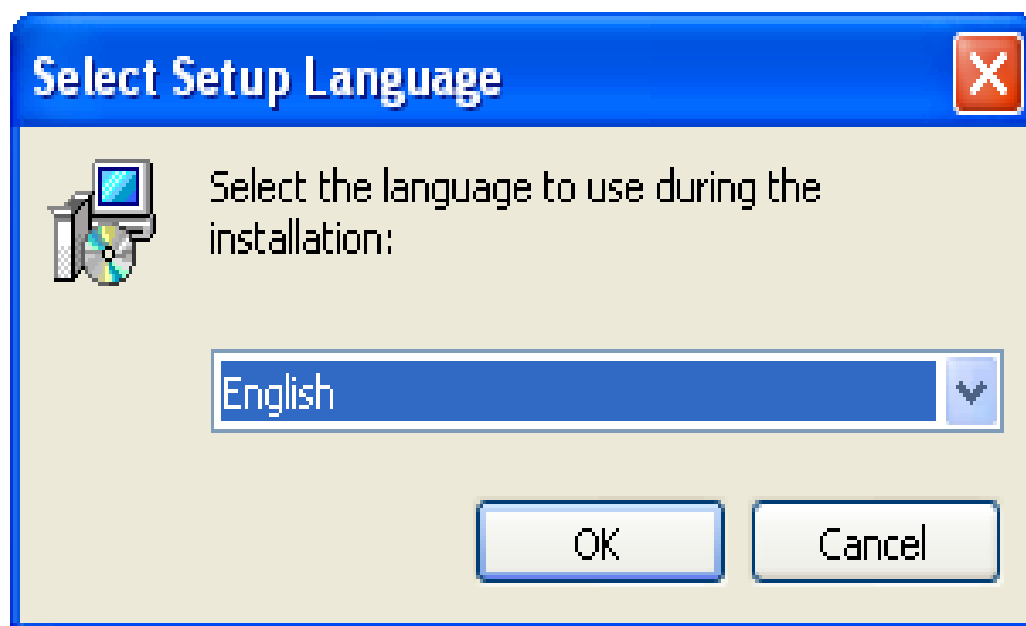


Figura 24

Presionar siguiente en todas las ventanas que se presenten posteriormente, de esta manera los archivos necesarios se instalará en la ruta por defecto.

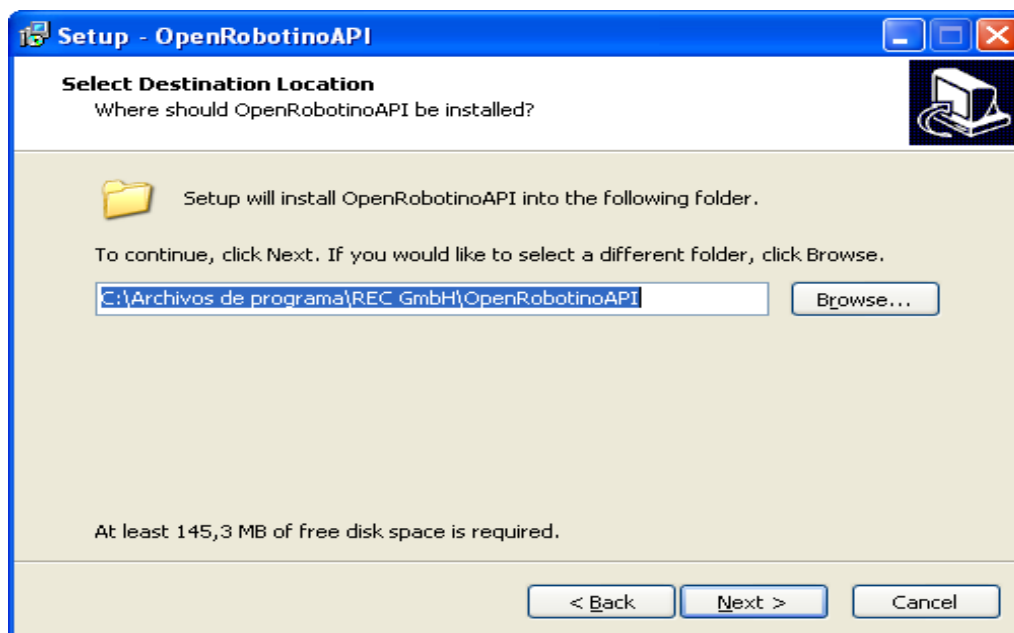


Figura 25

En la ventana Select Additional Tasks aparece una caja marcada, dejarlo como esta para añadir las aplicaciones contenidas del API en la ruta por defecto, luego presionar siguiente e instalarlo.

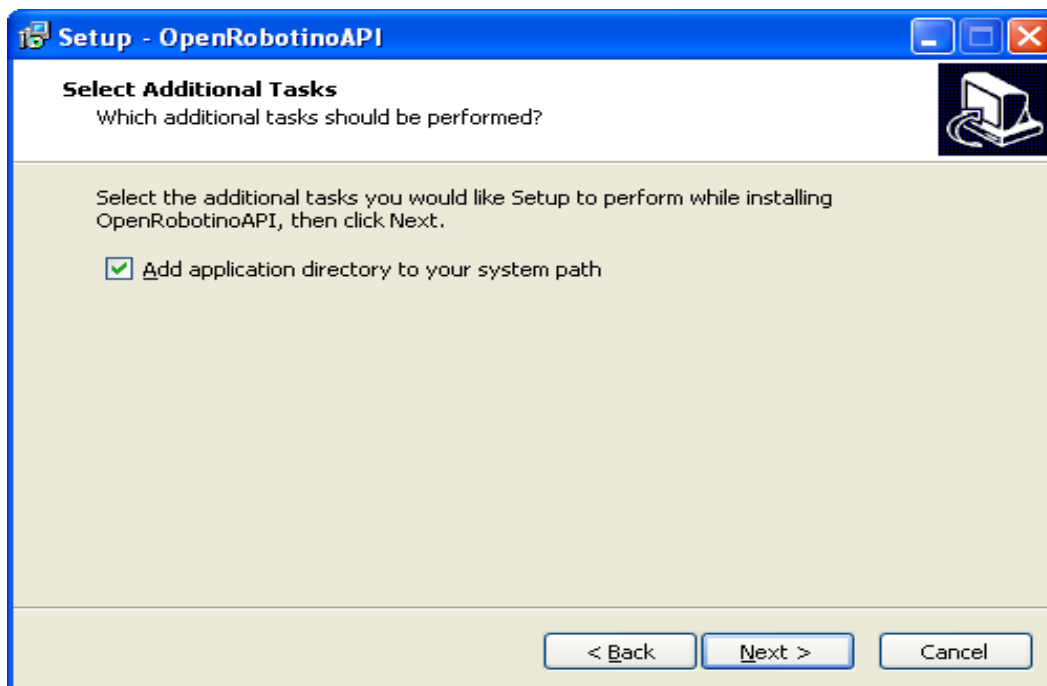


Figura 26

AGREGANDO REFERENCIA EN C# .NET 2010

Crear una nueva aplicación bajo C# .NET

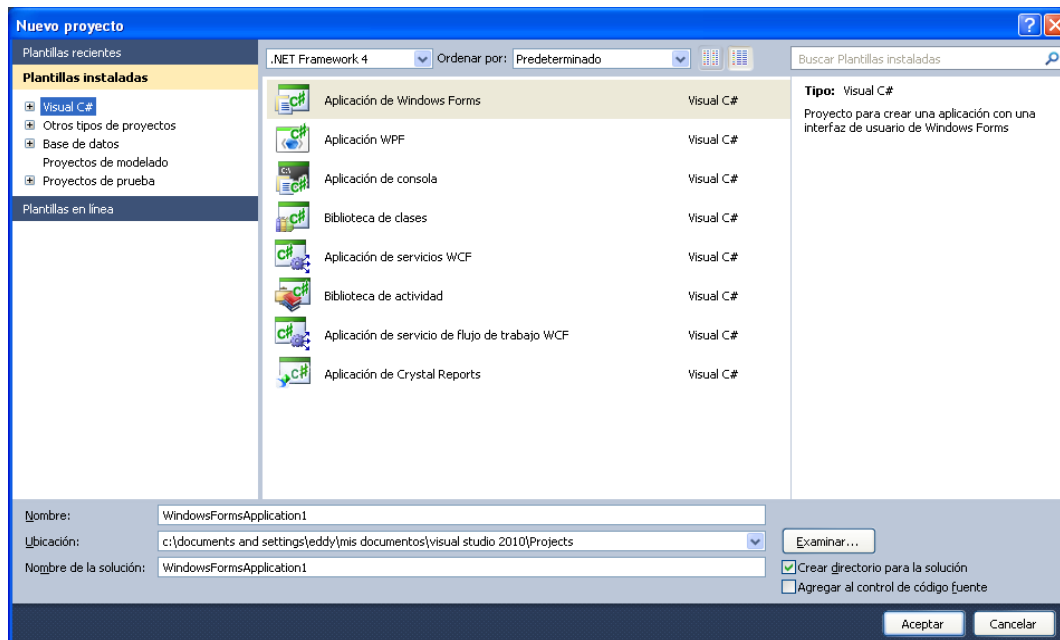


Figura 27

Abrir el explorador de soluciones y dar click derecho sobre las referencias de la aplicación, eligiendo la opción agregar referencia.

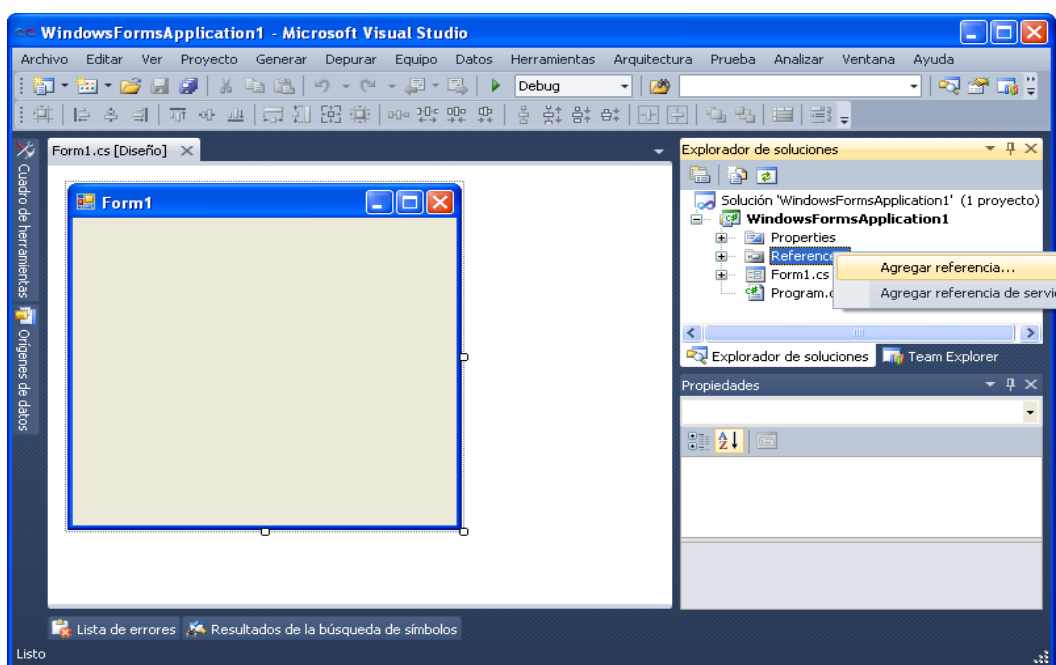


Figura 28

En la ventana que aparece ubicarse en examinar y buscar el directorio donde se instaló el API del Robotino.

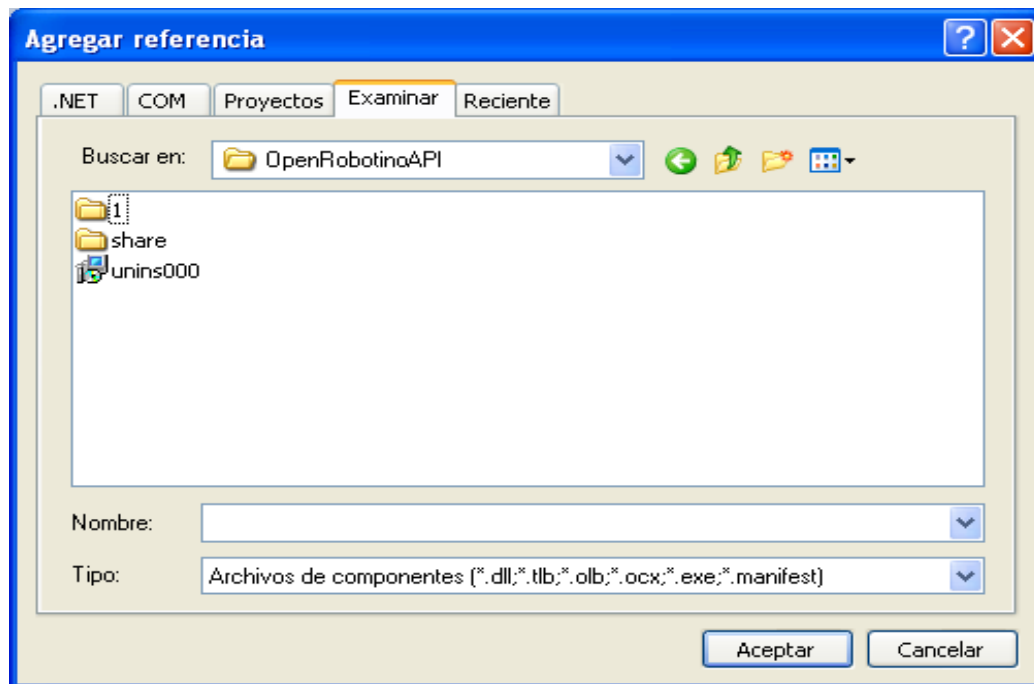


Figura 29

Una vez ubicado en el directorio adentrarse tres carpetas más, 1\lib\dotnet con el fin de encontrar la librería `rec_robotino_com_wrap.dll` que será necesaria para programar Robotino desde C# .NET.

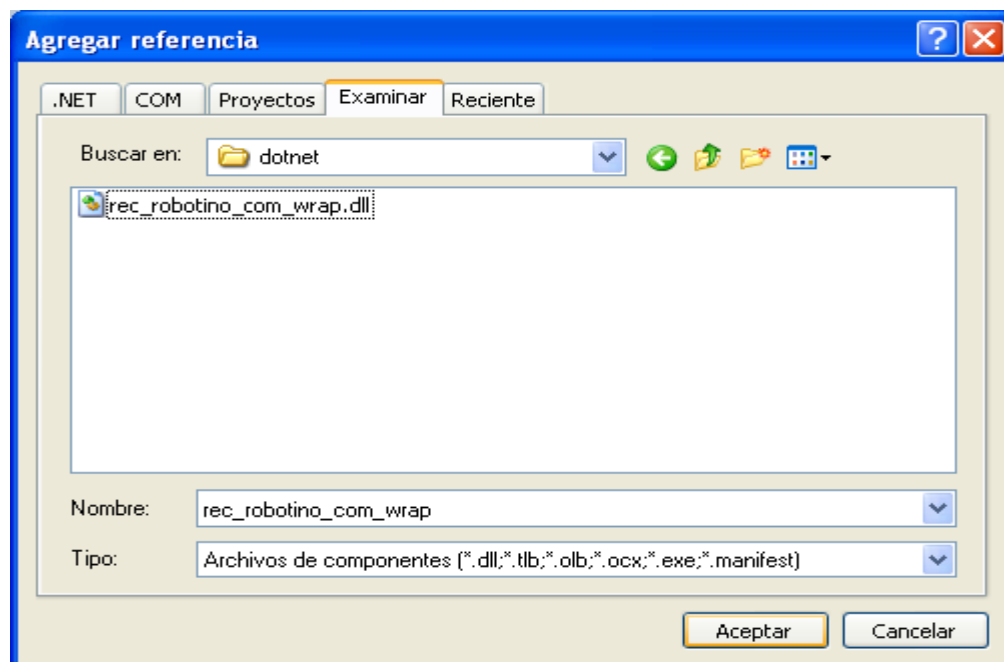


Figura 30

Crear una clase llamada Robot. En esta clase se realizarán las instancias necesarias de los componentes del Robotino.

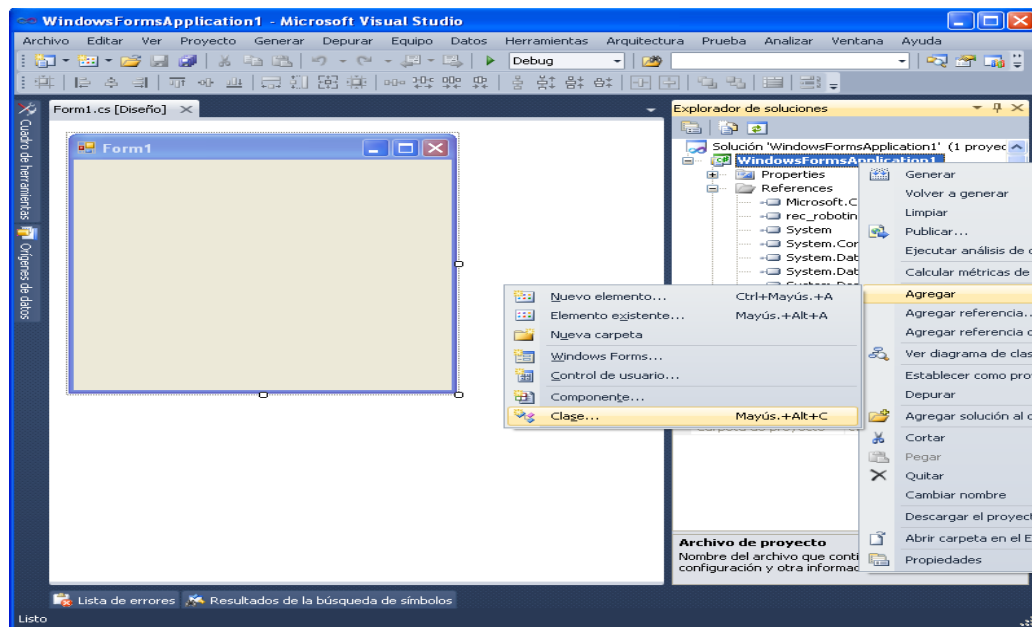


Figura 33

En la clase Robot agregamos la directiva `using rec.robotino.com;` con lo cual se habilitan las funciones de la referencia del Robotino, y la directiva `using System.Drawing;` para trabajar con imágenes.

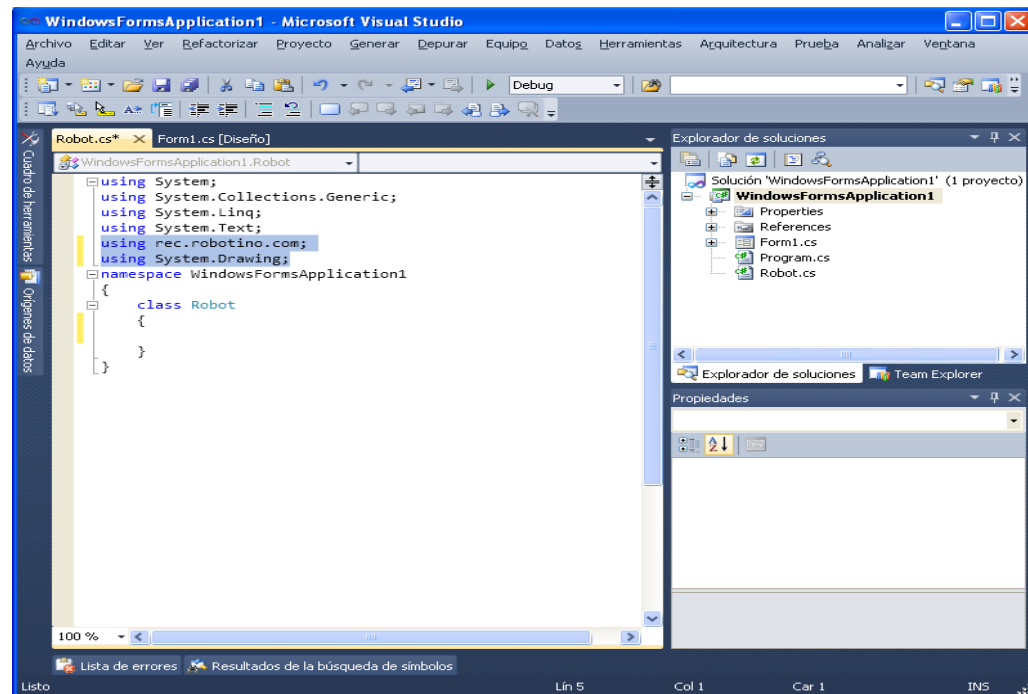


Figura 34

Dentro de la clase Robot, agregamos las siguientes líneas de comando:

- 1) Cuatro eventos que permiten a la aplicación conectarse, desconectarse, controlar errores y obtener la imagen de la Webcam del Robotino.

```
public delegate void ConnectedEventHandler(Robot sender);
public delegate void DisconnectedEventHandler(Robot sender);
public delegate void ErrorEventHandler(Robot sender, rec.robotino.com.Com.Error error);
public delegate void ImageReceivedEventHandler(Robot sender, Image img);

public event ConnectedEventHandler Connected;
public event DisconnectedEventHandler Disconnected;
public event ErrorEventHandler Error;
public event ImageReceivedEventHandler ImageReceived;
```

- 2) Definir variables para los componentes del Robotino en la aplicación, en este caso para la conexión, movimiento omnidireccional, para-choques y cámara.

```
protected readonly Com com;
protected readonly OmniDrive omniDrive;
protected readonly Bumper bumper;
protected readonly Camera camera;
```

Para instanciar otros componentes, el proceso se realiza de la misma manera.

- 3) Definir una variable para verificar el estado de la conexión

```
private volatile bool isConnected;
```

- 4) Crear un constructor de la clase Robot que inicialice las variables en el momento en que se crean.

```
public Robot()
{
    com = new MyCom(this);
    omniDrive = new OmniDrive();
    bumper = new Bumper();
    camera = new MyCamera(this);

    omniDrive.setComId(com.id());
    bumper.setComId(com.id());
    camera.setComId(com.id());
}
```

- 5) Agregar cuatros métodos en la clase. Estos permiten verificar la conexión, habilitar o deshabilitar la transmisión de la imagen, verificar estado del para-choques, conectar la aplicación con el Robotino y definir velocidades en el movimiento omnidireccional.

```

public bool IsConnected
{
    get
    {
        return isConnected;
    }
}
public bool CameraStreaming
{
    get
    {
        return camera.isStreaming();
    }
    set
    {
        camera.setStreaming(value);
    }
}
public bool IsBumper()
{
    return bumper.value();
}
public virtual void Connect(String hostname, bool blockUntilConnected)
{
    com.setAddress(hostname);
    com.connect(blockUntilConnected);
    Console.WriteLine("Connecting...");
}
public virtual void SetVelocity(float vx, float vy, float omega)
{
    omniDrive.setVelocity(vx, vy, omega);
}

```

6) Agregar dos clases heredadas de Com y Camera.

La clase MyCom posee los cuatro eventos mencionados en el paso 1, que a su vez pueden ser llamados desde otras clases cuando sean necesarios, con el fin de verificar constantemente el estado de la conexión del Robotino en la aplicación desarrollada bajo C# .Net.

```

private class MyCom : Com
{
    Robot robot;
    public MyCom(Robot robot)
    {
        this.robot = robot;
    }
    public override void connectedEvent()
    {
        Console.WriteLine("Connected");
        robot.isConnected = true;
        if (robot.Connected != null)
            robot.Connected.BeginInvoke(robot, null, null);
    }
    public override void connectionClosedEvent()
    {
        Console.WriteLine("Disconnected");
        robot.isConnected = false;
        if (robot.Disconnected != null)
            robot.Disconnected.BeginInvoke(robot, null, null);
    }
    public override void errorEvent(Error error, String errorStr)
    {
        Console.WriteLine("Error occurred: " + error);
        if (robot.Error != null)
            robot.Error.BeginInvoke(robot, error, null, null);
    }
}

```

La clase MyCamera posee el evento para invocar la imagen desde la Webcam del Robotino, con el fin de aplicar en ella técnicas de procesamiento de imágenes.

```

private class MyCamera : Camera
{
    Robot robot;

    public MyCamera(Robot robot)
    {
        this.robot = robot;
    }

    public override void imageReceivedEvent(Image data, uint dataSize, uint width, uint height_,
        uint numChannels, uint bitsPerChannel, uint step)
    {
        if (robot.ImageReceived != null)
            robot.ImageReceived.BeginInvoke(robot, data, null, null);
    }
}

```

Una vez creada la clase Robot y agregada todas las líneas de comando mencionadas anteriormente, ubicarse en el diseño de la clase Form y crear los siguientes componentes:

- 1 Textbox para ingresar la IP del Robotino.
- 1 PictureBox para mostrar la imagen de la Webcam.
- 1 Button para realizar la conexión.
- 7 Button para direccionamiento del Robotino en el entorno.
- 1 Button para verificar el estado del para-choques

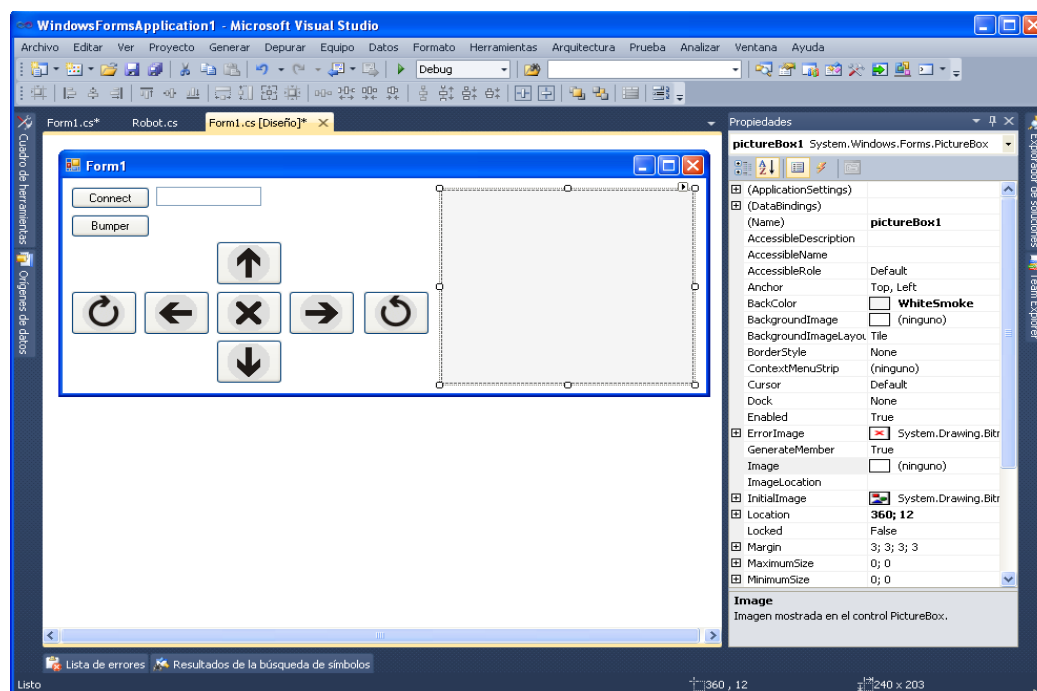


Figura 35

Ya agregado todos los componentes necesarios en el Form1, empezar a programar cada uno de los botones, para esto es necesario primeramente realizar una instancia de la clase Robot creada anteriormente de la siguiente manera `Robot rot = new Robot()`.

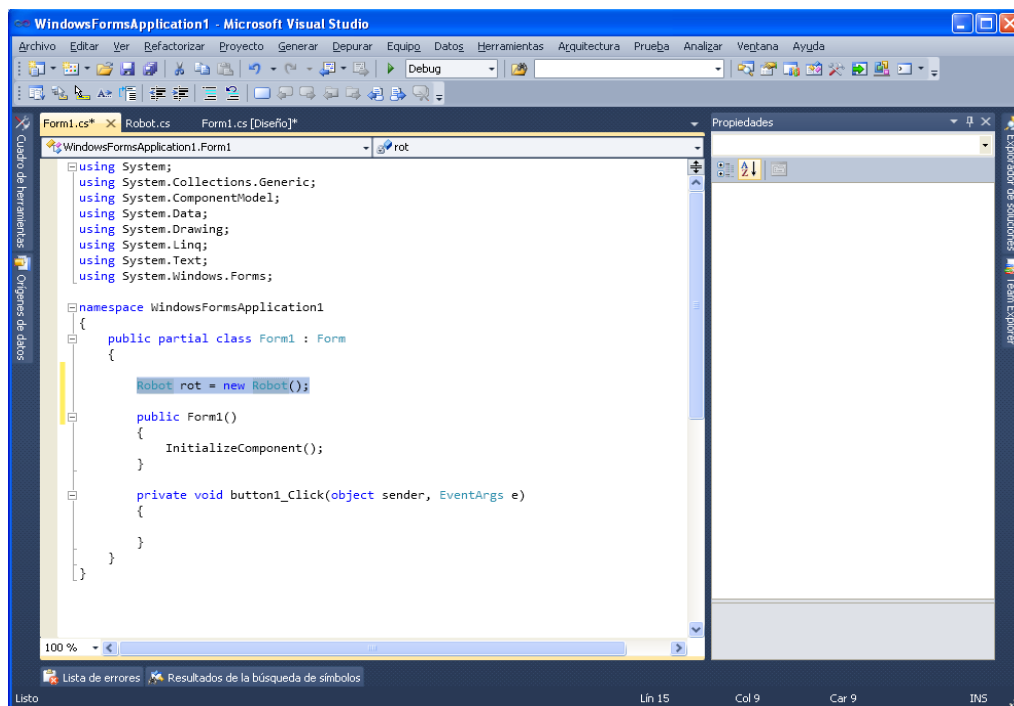


Figura 36

La siguiente tabla muestra las líneas de código necesario para cada botón, donde speed y rotSpeed son variables flotantes definidas al comienzo de la clase Form.

BOTÓN	CÓDIGO FUENTE
Conectar	rot.Connect(textBox1.Text,false);
Arriba	rot.SetVelocity(speed, 0.0f, 0.0f);
Abajo	rot.SetVelocity(-speed, 0.0f, 0.0f);
Izquierda	rot.SetVelocity(0.0f, speed, 0.0f);
Derecha	rot.SetVelocity(0.0f, -speed, 0.0f);
Giro a la derecha	rot.SetVelocity(0.0f, 0.0f, rotSpeed);
Giro a la izquierda	rot.SetVelocity(0.0f, 0.0f, -rotSpeed);
Para	rot.SetVelocity(0.0f, 0.0f, 0.0f);
Bumper	MessageBox.Show(rot.IsBumper().ToString());

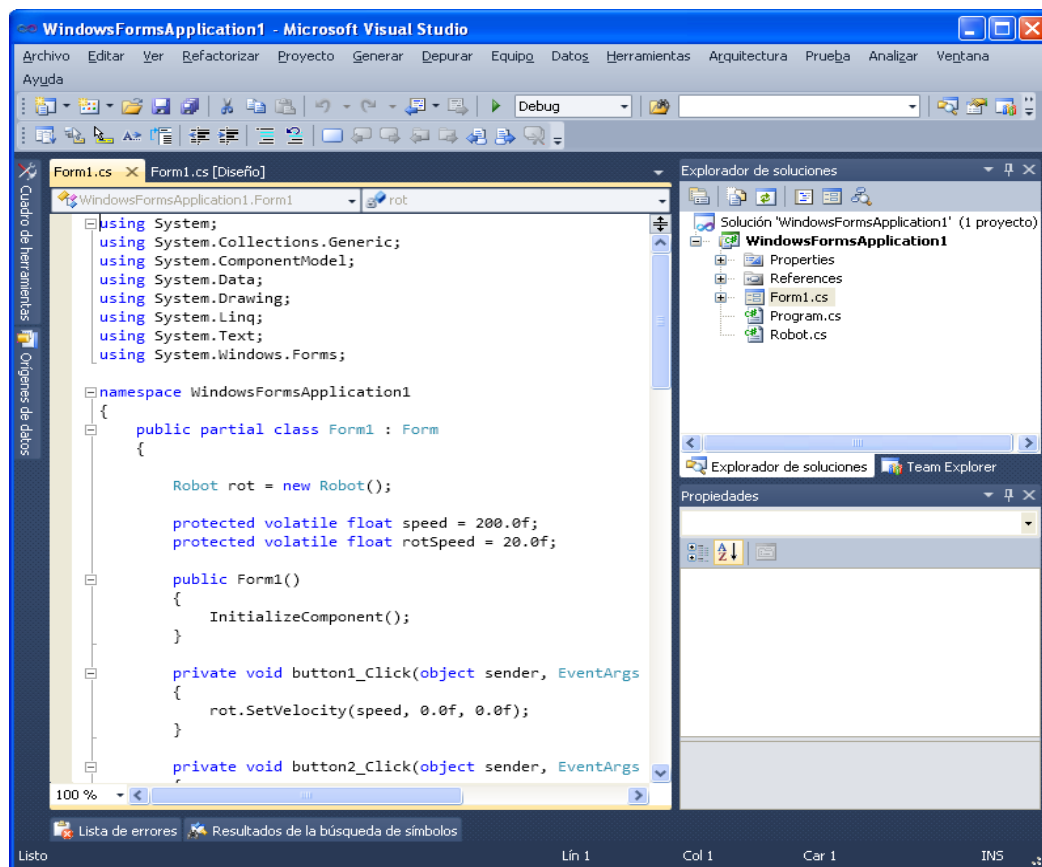


Figura 37

Con esto ya se logra controlar al Robotino desde la aplicación. Sin embargo, no se puede obtener la imagen de la Webcam, para aquello se requiere realizar un evento al cargar la clase Form, como también un método que actualice la imagen del picturebox constantemente.

```

private void Form1_Load(object sender, EventArgs e)
{
    rot.ImageReceived += new Robot.ImageReceivedEventHandler(robot_ImageReceived);
    rot.CameraStreaming = true;
}

void robot_ImageReceived(Robot sender, Image img)
{
    pictureBox1.Image = img;
    if (this.InvokeRequired)
        this.Invoke(new MethodInvoker(Invalidate));
}

```

La aplicación se puede ejecutar utilizando el simulador en 3D desarrollado para el Robotino, denominado Robotino® SIM Demo con la IP 127.0.0.1:8080.

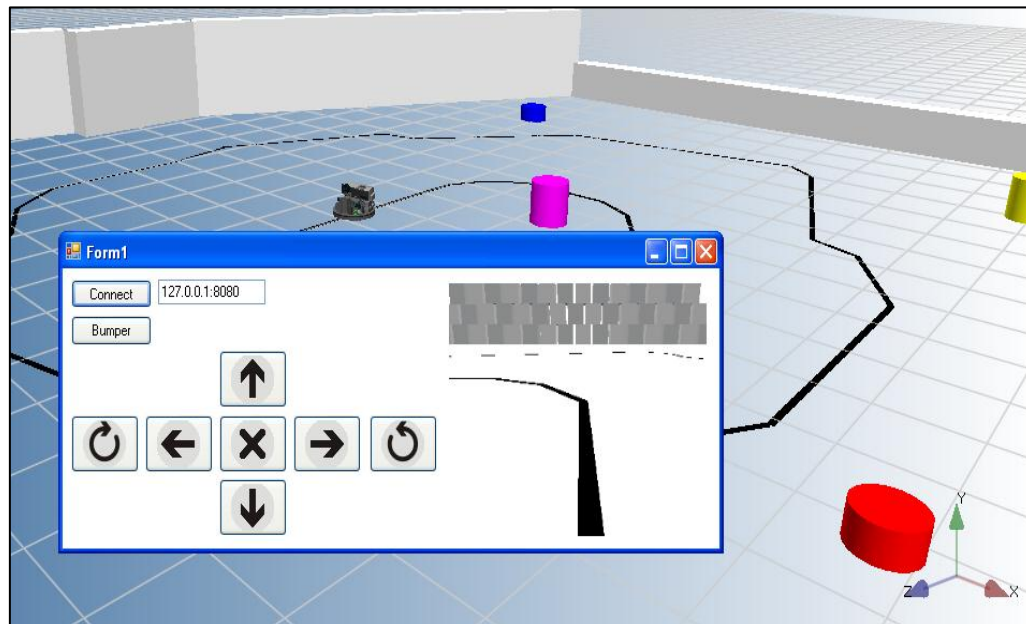


Figura 38

Para ejecutar la aplicación con el Robotino de FESTO en un entorno real se debe realizar primeramente una conexión de red entre la PC y la red del Robotino. Esto se logra ubicándose en *Inicio - Panel de control – Conexiones de redes* (Windows XP).

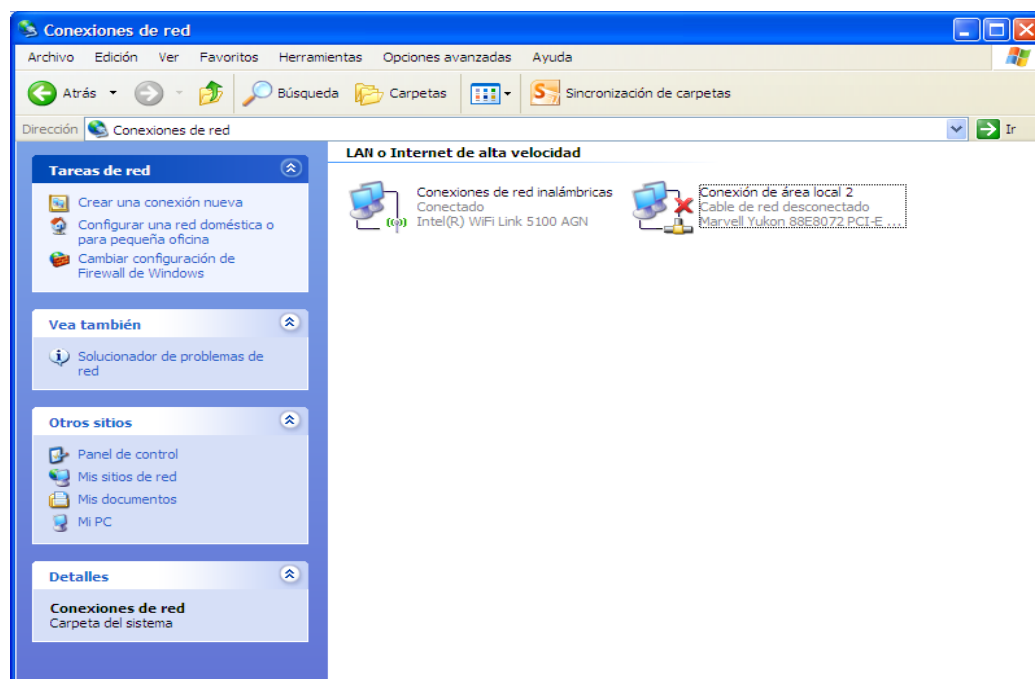


Figura 39

Sobre el icono de conexiones de red inalámbricas, dar click derecho y seleccionar *ver redes inalámbricas*, en aquella ventana que aparece buscar la red del Robotino y presionar en el botón conectar.

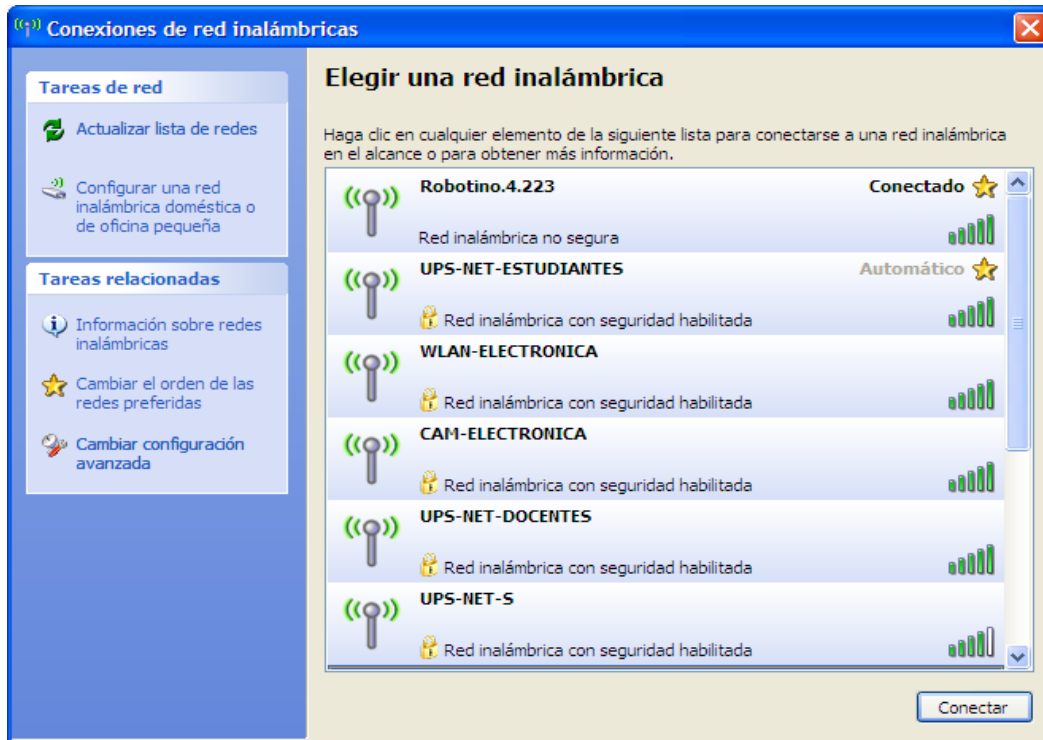


Figura 40

Una vez realizada la conexión de red, ejecutar la aplicación y cambiar la IP del simulador por la IP del robot móvil Robotino de FESTO que se tenga en ese momento, con el fin de observar su funcionamiento en tiempo real.

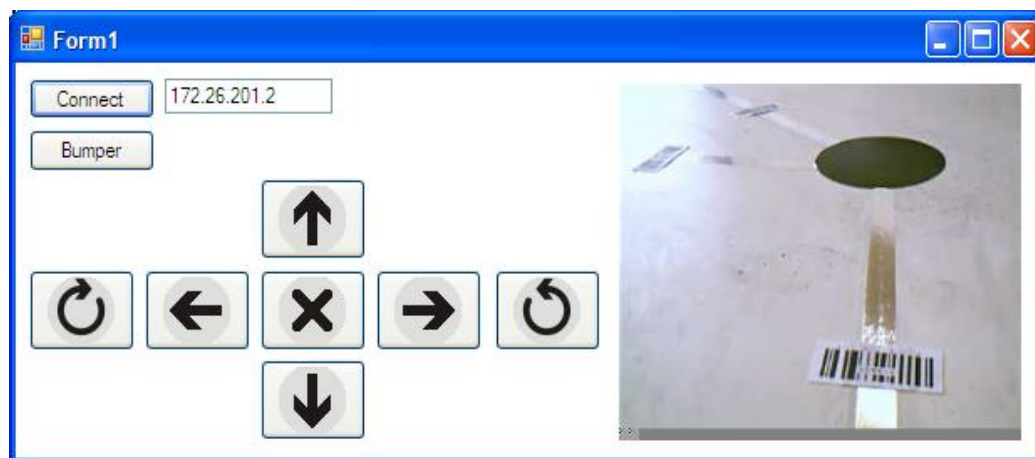


Figura 41

INICIO DEL SISTEMA DE BÚSQUEDA

Cuando la aplicación es iniciada, se visualiza la ventana principal donde se puede ejecutar la búsqueda haciendo uso del entorno virtual o utilizando el Robotino de FESTO en tiempo real.

A continuación se detallan los botones utilizados en el sistema de búsqueda informada.

Botones de acceso directo en la ventana principal



Permite conectar la aplicación con el Robotino de FESTO



Ejecuta la búsqueda en el entorno virtual o utilizando el robot móvil en un entorno real.



Pausa la aplicación cuando se está ejecutando el proceso de búsqueda.



Detiene el proceso de búsqueda.



Activa el proceso de búsqueda en modo Robotino cuando existe conexión de red hacia él.



Activa el proceso de búsqueda en modo simulador en el mismo entorno del sistema.



Permite crear un nuevo entorno virtual para el proceso de búsqueda.



Permite editar las configuraciones necesarias para el funcionamiento del Robotino en el entorno de trabajo, como también activa los botones de crear y eliminar nodos o arcos.



Permite crear los nodos en el entorno virtual.



Permite crear los arcos o enlaces entre nodos.



Elimina nodos o arcos creados en el entorno virtual.

Botones de acceso en el menú principal del sistema



Permite cargar entornos virtuales almacenados en archivos XML con todas sus configuraciones.



Guarda archivos en formato XML con todas las configuraciones del entorno virtual.



Cierra la aplicación de búsqueda.



Accede a las configuraciones del Robotino cuando se tiene una conexión de red establecida entre la PC y el Robotino.



Accede al modo teleoperado del Robotino en el entorno de trabajo.



Accede al análisis de resultados generados por el proceso de búsqueda.



Muestra un manual de usuario del sistema de búsqueda.

➤ Creación de un entorno virtual

Presionar sobre el botón *nuevo entorno*, y en la ventana que aparece definir el ancho y alto del entorno virtual a crear. Hay que tomar en cuenta que estos datos son definidos en centímetros, utilizados cuando se requiere aplicar la búsqueda en un entorno real.

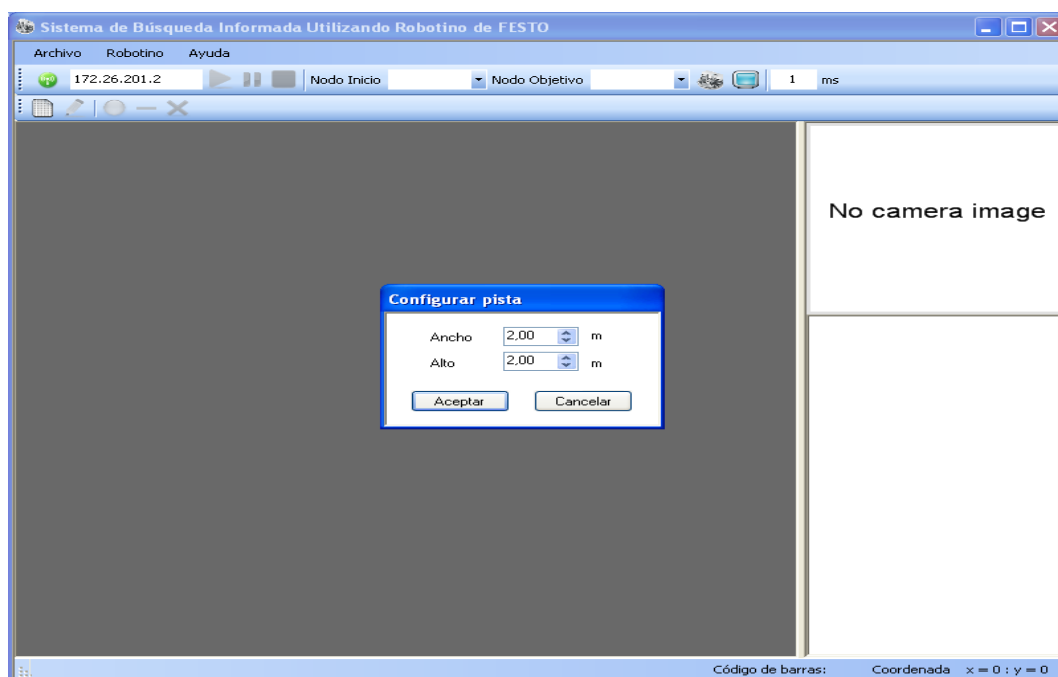


Figura 42

Una vez definido el tamaño del entorno, presionar sobre el botón *crear nodos*, luego ubicar el puntero del mouse en cualquier parte del entorno virtual y dar click en el lugar que se desee graficar el nodo.

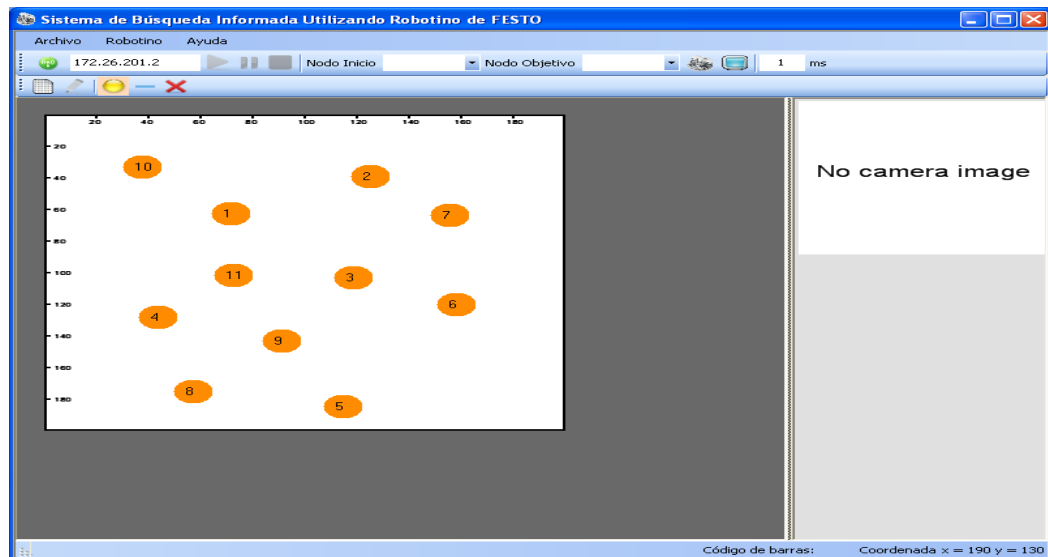


Figura 43

Después de crear los nodos en el entorno virtual, crear los enlaces que se consideren necesarios para observar su funcionamiento. Esto se logra presionando sobre el botón *crear arcos*, luego ubicar el puntero del mouse sobre cualquier nodo y dar click sobre él, después ubicar el puntero sobre otro nodo con el fin de crear un enlace o arco entre ellos.

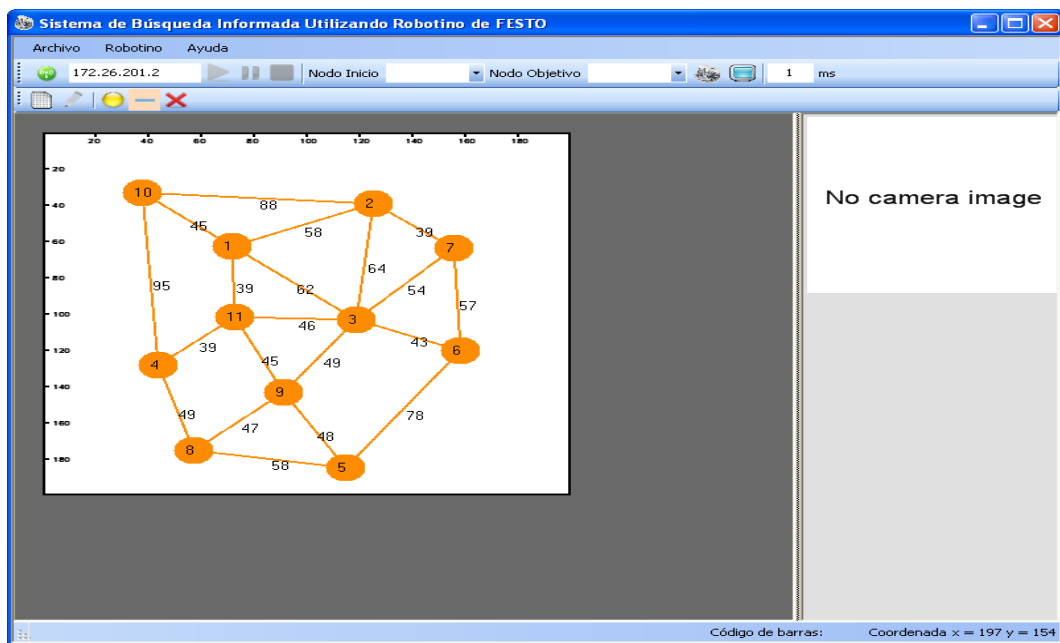


Figura 44

Nota: Los valores de los nodos y arcos son generados automáticamente por el sistema de búsqueda, no requieren de configuración alguna.

Si se desea eliminar algún nodo o arco que no esté correctamente ubicado, presionar sobre el botón eliminar, luego con el mouse ubicarse sobre el nodo a eliminar y darle click derecho. Esto muestra al nodo seleccionado y todos los enlaces hacia él, si se desea eliminamos todos los enlaces seleccionar la primera opción, caso contrario seleccionar el enlace que se desee eliminar.

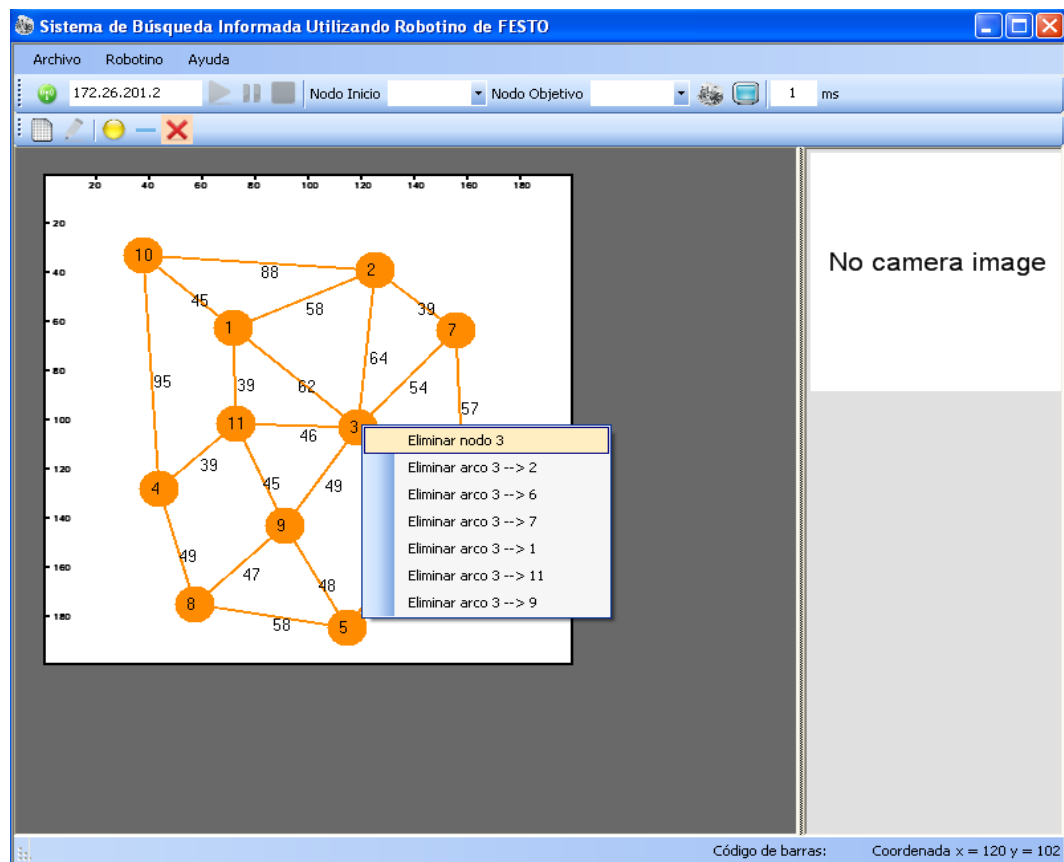


Figura 45

Ya creados los arcos y nodos en el entorno virtual, definir un nodo inicial y un nodo objetivo en la ventana principal de la aplicación, luego presionar sobre el botón simulador para ejecutar la aplicación sin el Robotino. También se puede definir el tiempo del proceso de búsqueda para visualizar su funcionamiento más detenidamente.

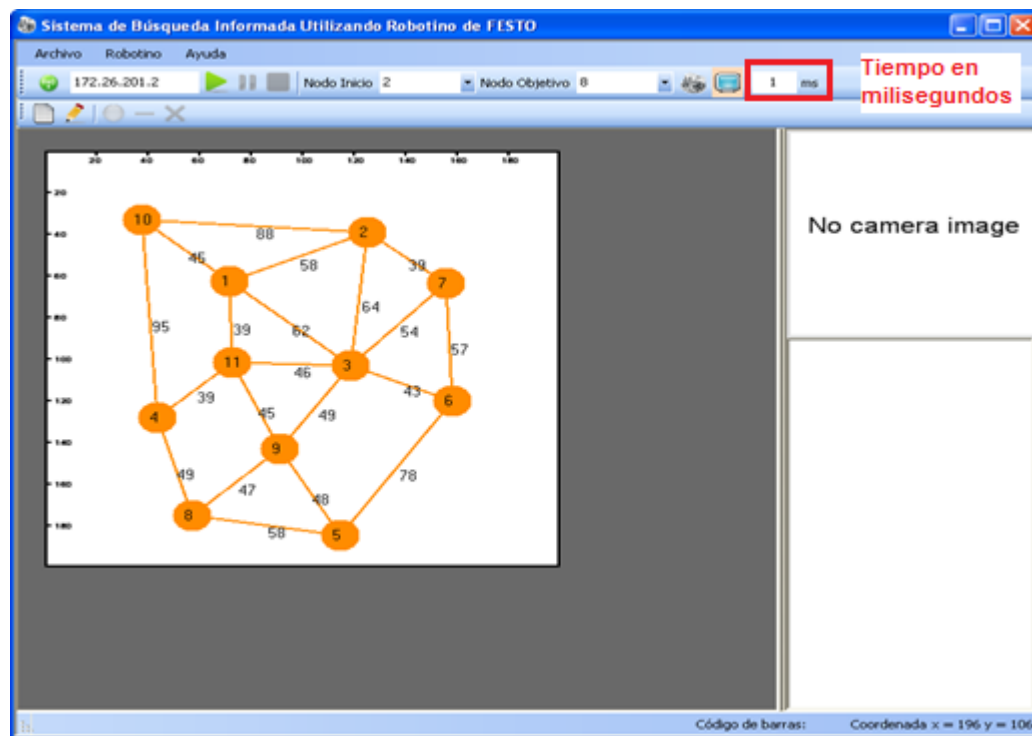






Figura 46

Una vez definidos todos los parámetros, presionar el botón ejecutar, con el fin de observar su funcionamiento y resultados. En la siguiente tabla se detalla el significado de cada color mostrados en la figura 25

Color	Detalle
	Nodos óptimos generados por el algoritmo A*.
	Nodos que no han sido considerados como óptimos pero que han sido analizados por el algoritmo de búsqueda.
	Nodos próximos a ser analizados por el algoritmo de búsqueda.
	Nodos que no han sido tomados en cuenta durante el proceso de búsqueda.

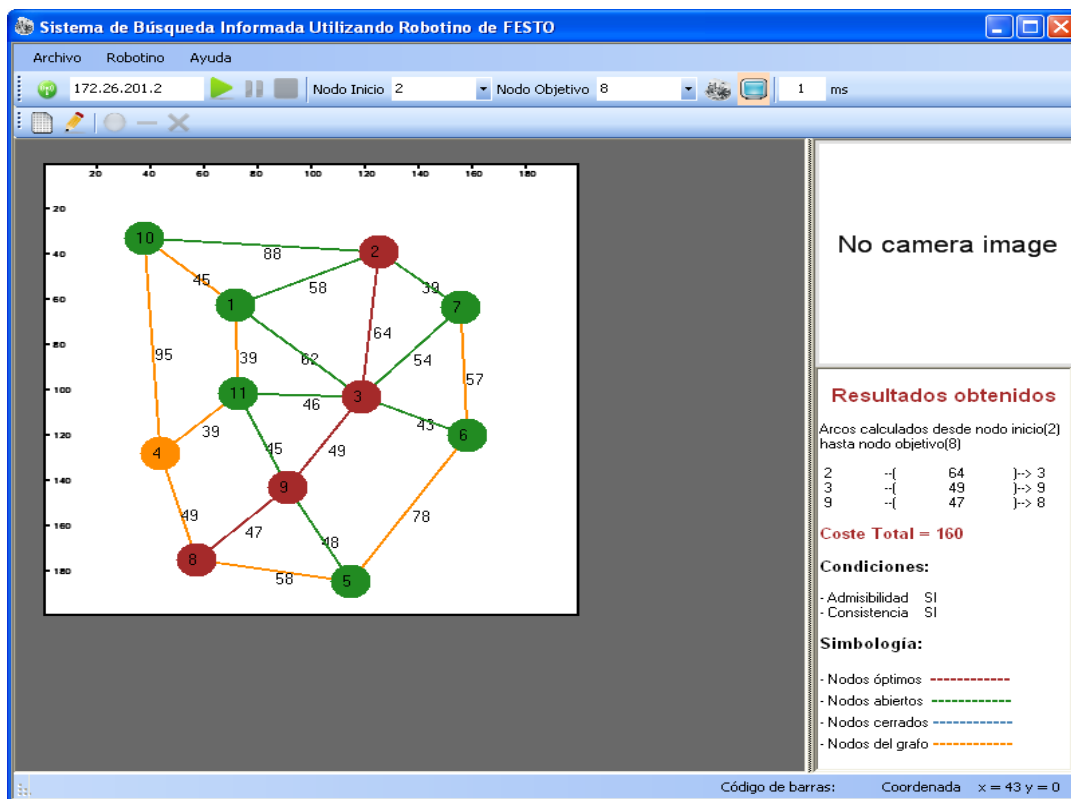


Figura 47

Ejecutando aplicación de búsqueda en tiempo real

Crear un entorno virtual con arcos y nodos, los cuales deben ser construidos en el entorno de trabajo como se muestra en la figura 26.

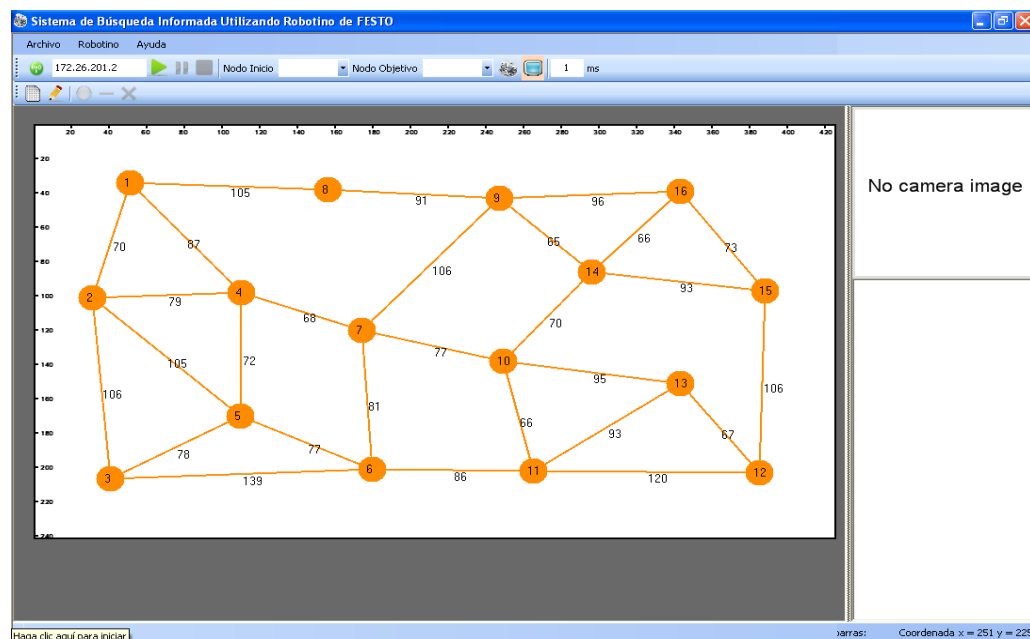


Figura 48

Para empezar con la configuración presionar sobre el botón *edita entorno*, a lado derecho se presenta una tabla que contiene todos los enlaces de los nodos. Sin embargo, la fila código de barras de la tabla se encuentra con valores nulos, en esta parte es donde se registran todos los códigos de barra (Code 128) necesarios para la ejecución con el Robotino.

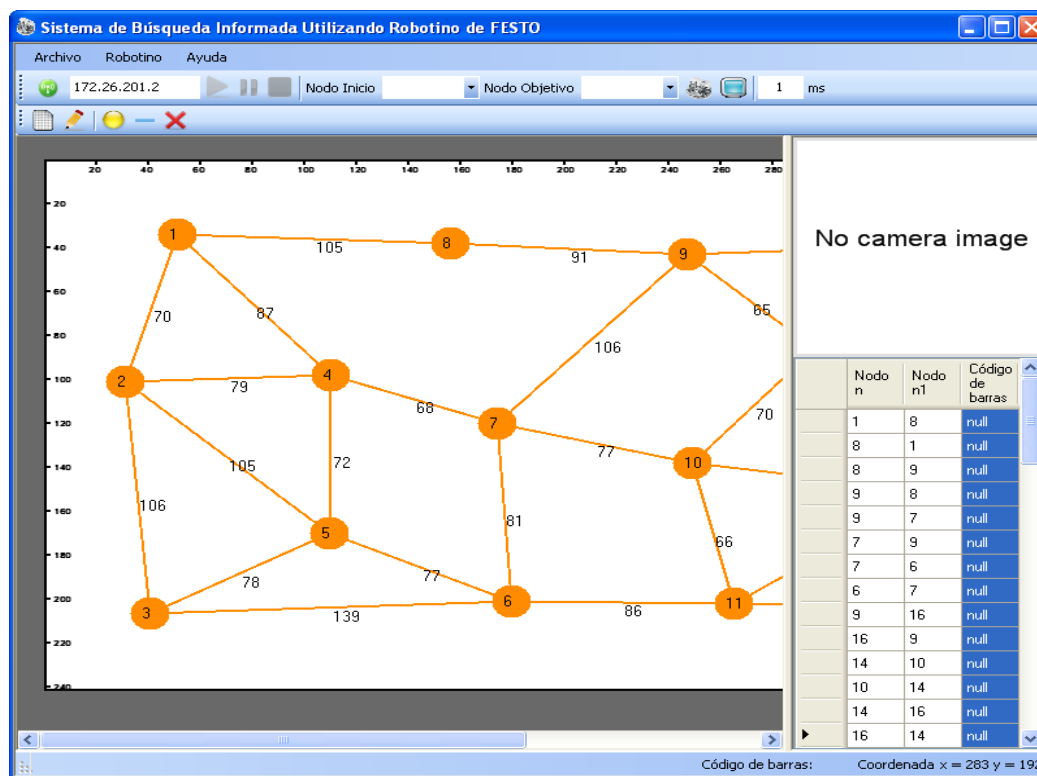


Figura 49

Para generar los códigos de barras se puede utilizar cualquier aplicación que genere códigos tipo CODE 128, en este caso se ha utilizado la aplicación Demo VintaSoft Barcode .NET SDK disponible en la dirección http://download.cnet.com/VintaSoft-Barcode-NET-SDK/3000-2070_4-10669290.html, una vez instalado la aplicación nos dirigimos a su directorio, normalmente archivos de programa, luego en la carpeta examples ejecutamos la aplicación para C# .Net. Ejecutando la aplicación nos ubicamos en Write, luego en la opción **barcode** seleccionamos Code 128 y en la opción **value** ingresamos el número del código de barra a generar. El mismo que será ingresado manualmente en el sistema de búsqueda.

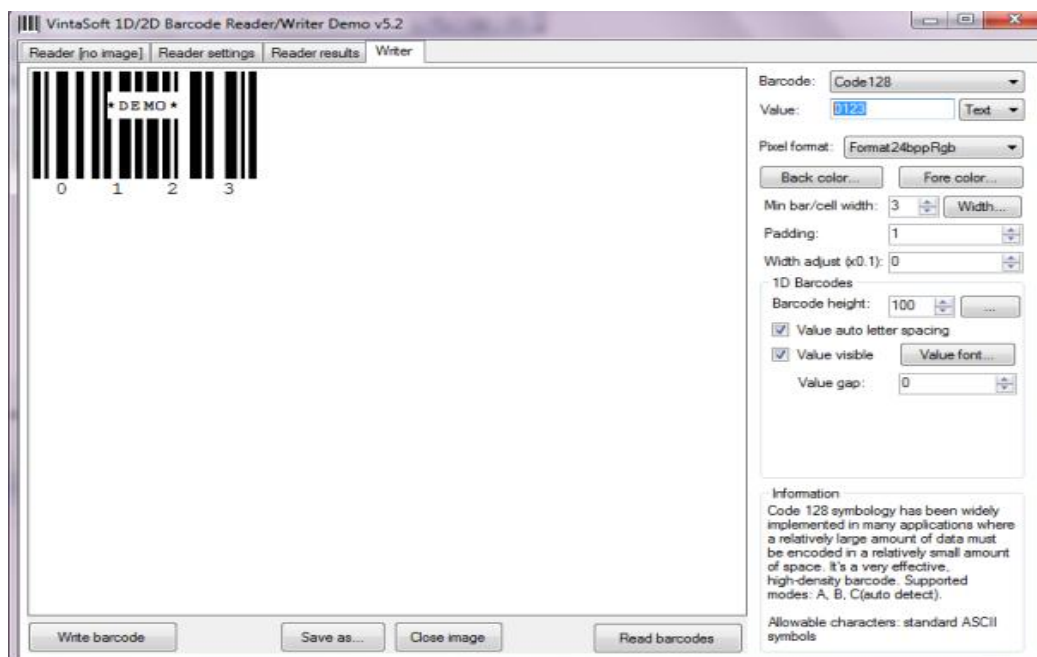


Figura 50

Nota: El sistema omite el primer dígito del código de barras, razón por la cual es necesario que el primer dígito sea cero.

Finalmente presionar sobre *Write barcode*, guardarlo e imprimirlo. Estos códigos impresos deben ser ubicados en cada camino del entorno de trabajo para que el Robotino junto con el control del sistema de búsqueda logre identificarlo.



Figura 51

Una vez generados los códigos de barras necesarios, se procede a registrarlos en el sistema, tomando en cuenta que la separación en los caminos del entorno de trabajo iguales a 65cm y menores a 80cm requieren un solo código de barras, mientras que mayores a 80cm requieren de dos códigos de barras en cada camino.

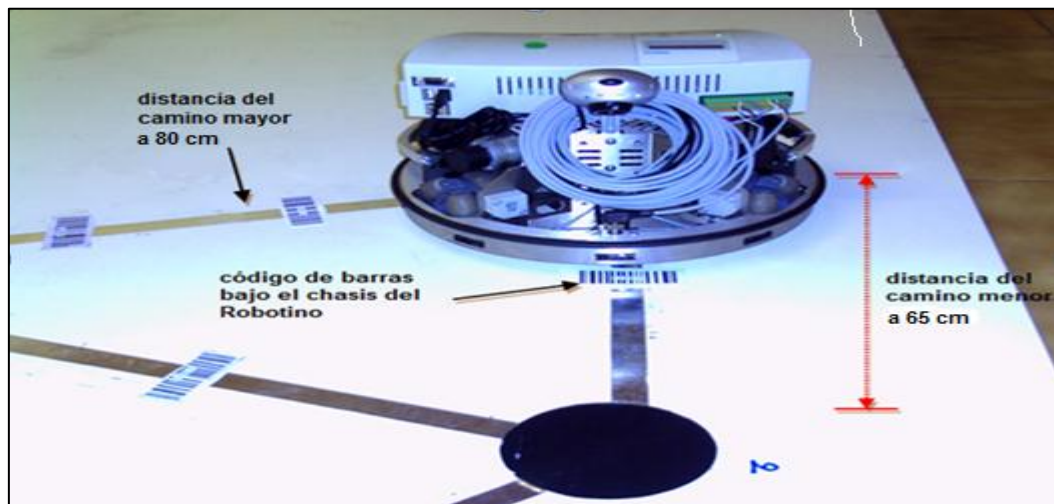


Figura 52

Cuando los códigos de barras son registrados en el sistema de búsqueda, son claramente visibles aquellos que utilizan un solo código en cada camino, ya que el mismo se repite para ambos lados. Por ejemplo, del nodo 10 al 14 y del nodo 14 al 10.

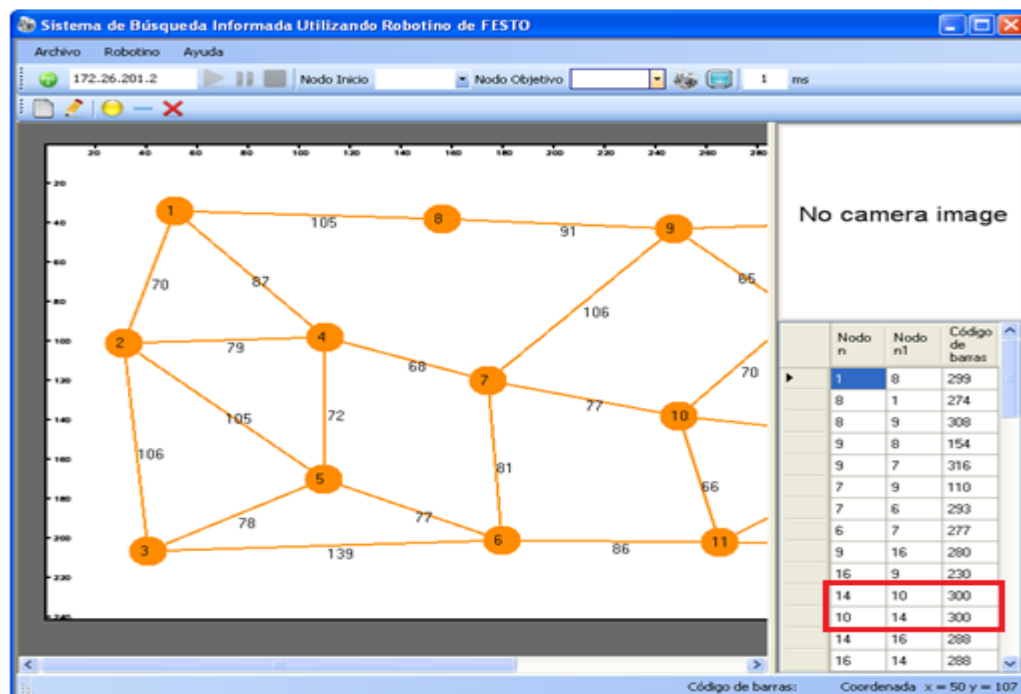


Figura 53

Ya registrados todos los códigos de barras, el siguiente paso es configurar el Robotino en *Menú – Robotino – Configuraciones* en el caso que no se desee trabajar con las configuraciones predeterminadas, siendo estas velocidad de los motores, giros, etc.

Parámetros

La ventana permite al usuario seleccionar dos sensores ópticos para la detección de nodos en el entorno de trabajo, un sensor análogo para la tarea de seguimiento por la cinta de aluminio y las configuraciones necesarias para:

- Configurar los límites del aluminio para la tarea de seguimiento.
- Escoger el motor para los datos del encoder.
- Configurar la distancia de recorrido del motor para capturar el código de barras al momento de detectar un camino.
- Configurar el valor máximo de centrado cuando el Robotino se encuentra en un nuevo nodo

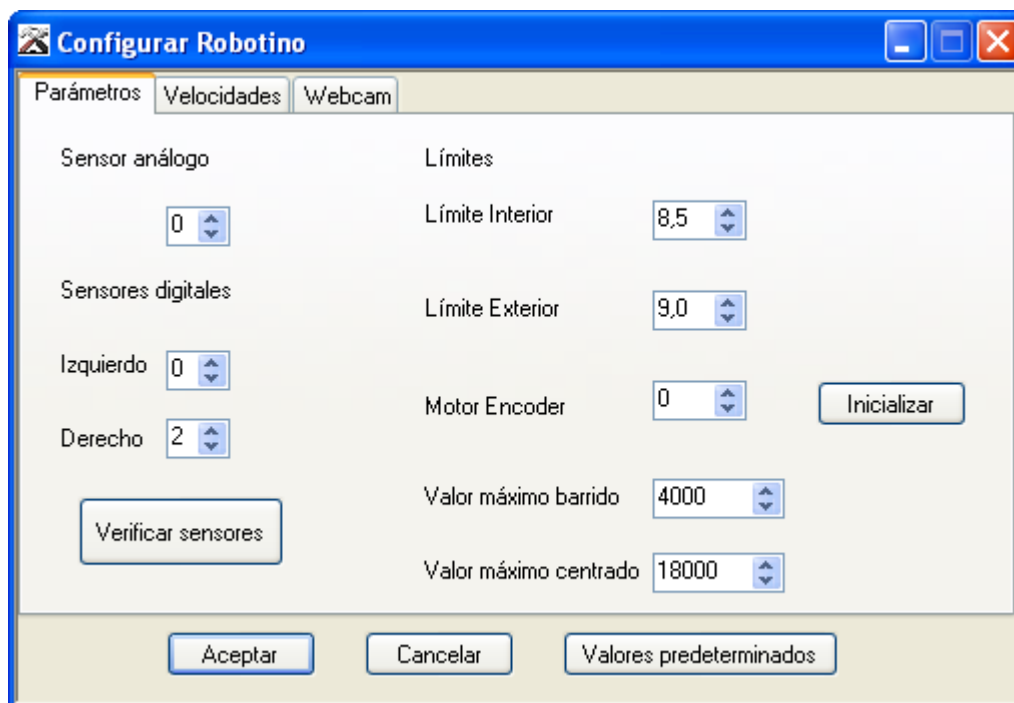


Figura 54

Velocidades

Aquí se logra configurar las velocidades de cada una de las tareas que realiza el Robotino en el entorno de trabajo, tales como seguimiento, barrido en la captura del código de barras, giro entre caminos, centrado en el nodo y posicionamiento para ir al siguiente nodo.

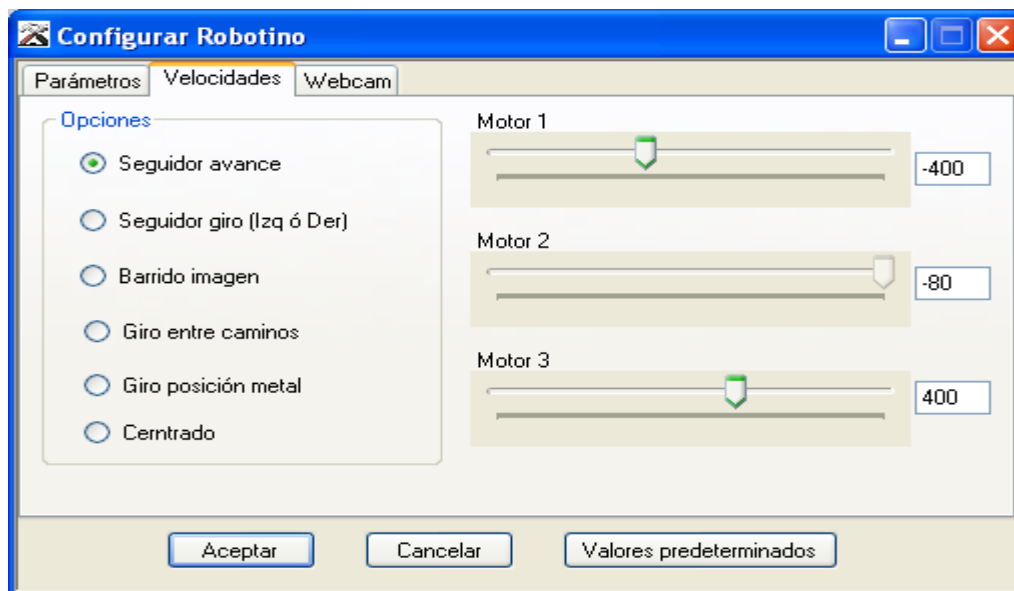


Figura 55

Webcam

Permite verificar la nitidez de la Webcam, identificando un código de barra en el entorno de trabajo.

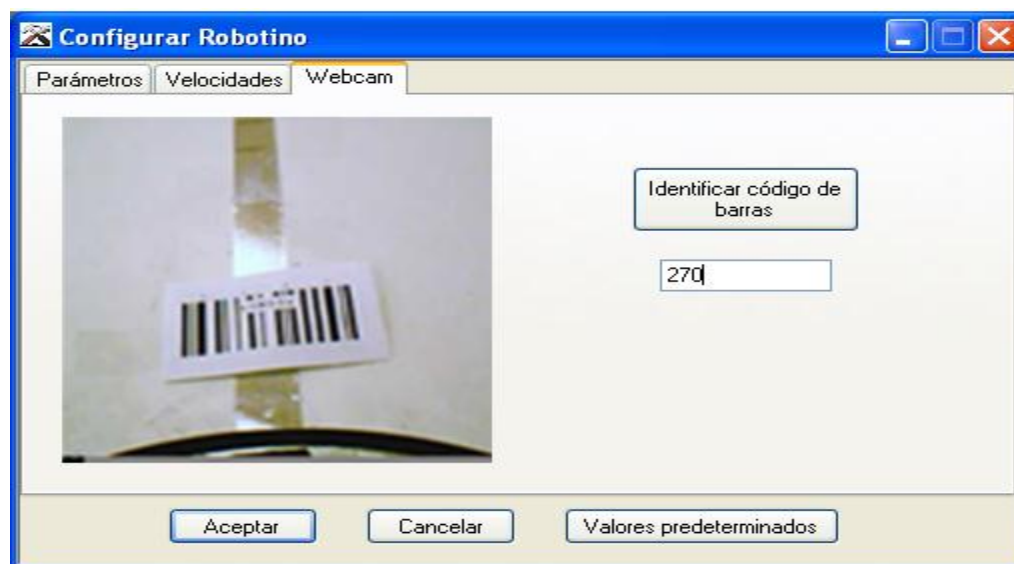


Figura 56

Ya configurados los respectivos parámetros en el sistema de búsqueda, se debe ubicar al Robotino sobre el nodo inicial utilizando el panel de control.

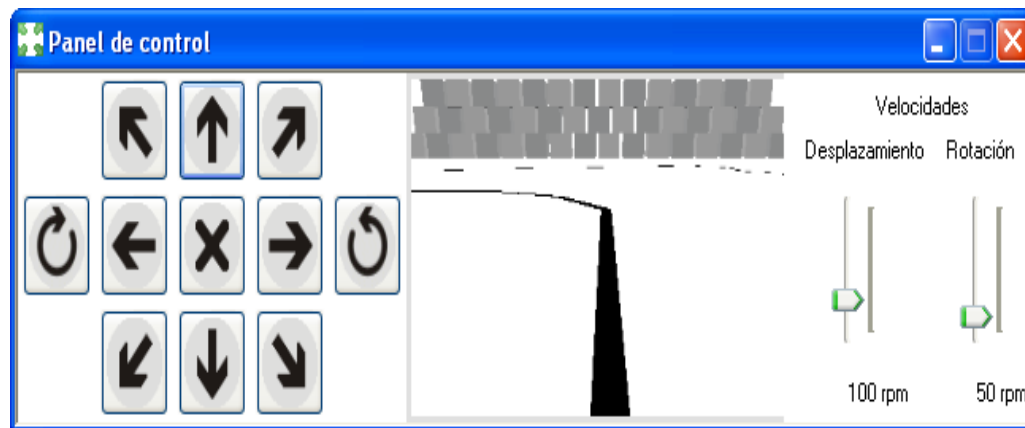


Figura 57

Luego presionar sobre el botón *ejecutar* para observar su funcionamiento en tiempo real. Como prueba se tomó un nodo inicial (16) y un nodo objetivo (3), presentando los siguientes resultados.

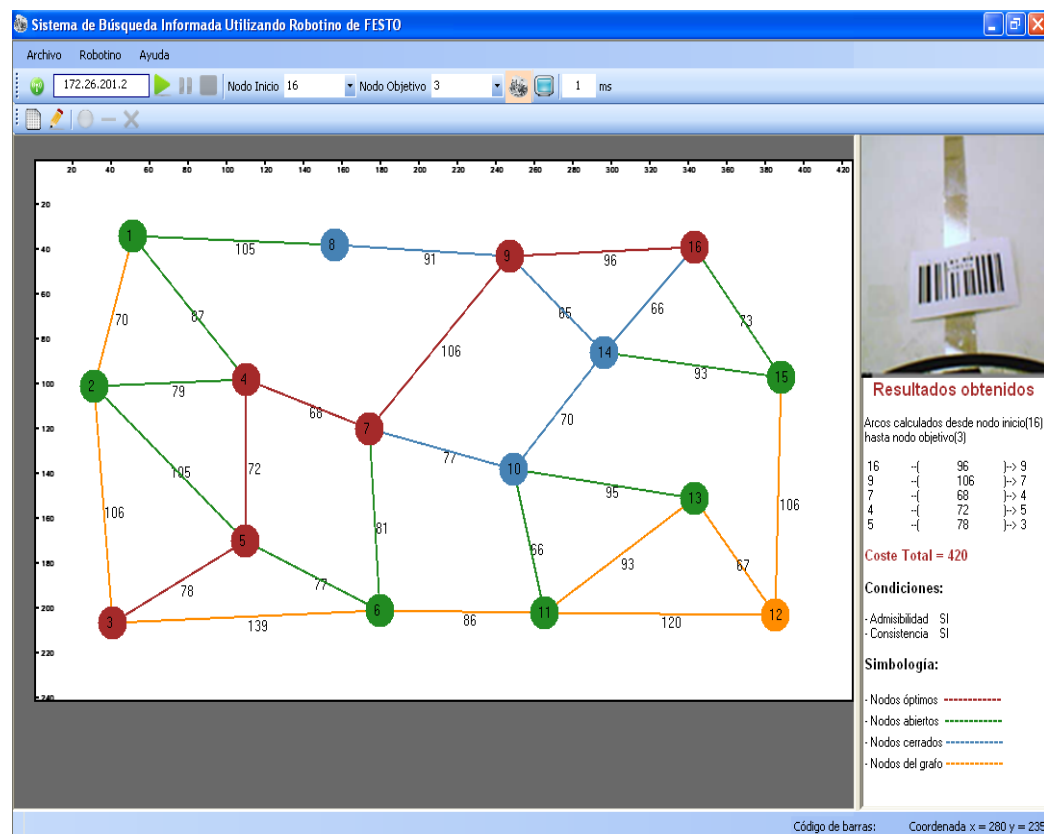


Figura 58

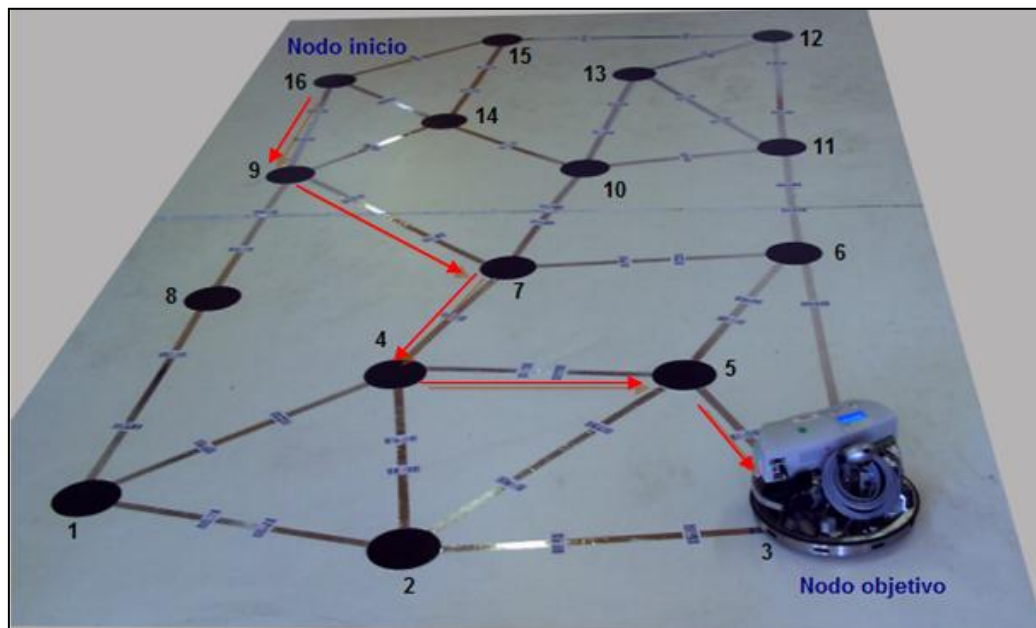


Figura 59

Para mayor comprensión del algoritmo de búsqueda A* se puede verificar los resultados generados al finalizar el proceso de búsqueda. Estos resultados muestran los procedimientos realizados por el algoritmo para encontrar el camino más óptimo. Sin embargo, los resultados pueden tardar a acuerdo a la cantidad de nodos analizados durante el proceso de búsqueda.

Resultados

RESULTADOS

ESPACIO DE ESTADOS							LISTA ABIERTOS		LISTA CERRADOS	
Ni	Ni	Nivel(Ni)	g(Ni,Ni)	h(Ni)	g(Nini,Ni)	f(Ni)	Nodo n	f(n)	Nodo n	
NODO n = 16										
16	null	0	0	346	0	346	14	348	16	
9	---	16	1	96	263	96	9	359		
14	---	16	1	66	282	66	15	437		
15	---	16	1	73	364	73				
NODO n = 14										
16	null	0	0	346	0	346	10	355	16	
9	---	16	1	96	263	96	9	359	14	
14	---	16	1	66	282	66	15	437		
15	---	16	1	73	364	73				
10	---	14	2	70	219	136				
NODO n = 10										
16	null	0	0	346	0	346	9	359	16	
9	---	16	1	96	263	96	9	359		
14	---	16	1	66	282	66	15	437		
15	---	16	1	73	364	73				
10	---	14	2	70	219	136				

Figura 60

ANEXO 3

Documentación Técnica del Robotino

ENCODERS

Incremental Encoders for BLDC Motors Inkrementalgeber für BG-Motoren

As standard, brushless DC motors of the BG range are equipped with Hall sensors for measuring current motor speed. Where more stringent demands are placed on the quality of regulation and positioning accuracy, the motors are available with a digital incremental encoder.

Incremental encoders have no sliding contacts and are not subject to wear. A light-emitting diode, a slotted metal disc, and a photo-diode array form a photoelectric circuit.

An internal logic produces two square-wave signals phased at 90° to each other from the output of the photo-diodes, with or without a reference impulse.

Where the cable length between the motor and encoder is more than 2.5 m, we recommend use of the RE .. TI, fitted with an additional power booster.

The standard supply voltage for the incremental encoder is 5 VDC. As specials, 24V versions are also available.

An IP54 cover is recommended as protection against external influences. In combination with motor BG 65, the incremental encoder can be incorporated in the IP65 extruded motor body.



Bürstenlose Gleichstrommotoren der Baureihe BG sind standardmäßig mit Hallsensoren zur Erfassung des Drehzahl-Istwertes ausgestattet.

Bei erhöhten Anforderungen an die Regelbarkeit und Positioniergenauigkeit sind die Motoren zusätzlich mit einem digitalen Inkrementalgeber erhältlich.

Die Inkrementalgeber arbeiten berührungslos und verschleißfrei. Eine Leuchtdiode, eine metallische Schlitzscheibe und ein Fotodiodenarray bilden eine Lichtschranke. Eine interne Logik erzeugt aus dem Signal der Fotodioden zwei um 90° verschobene Rechtecksignale, ohne bzw. mit Referenzimpuls.

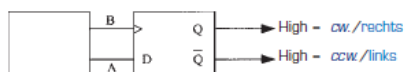
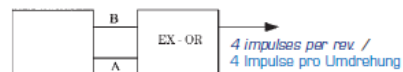
Bei Kabellängen von mehr als 2,5 m zwischen Motor und Geber empfiehlt sich der Einsatz eines RE .. TI, ausgerüstet mit einem zusätzlichen Leistungstreiber.

Die Versorgungsspannung der Inkrementalgeber beträgt standardmäßig 5 VDC. In Sonderausführungen sind auch 24V-Versionen erhältlich.

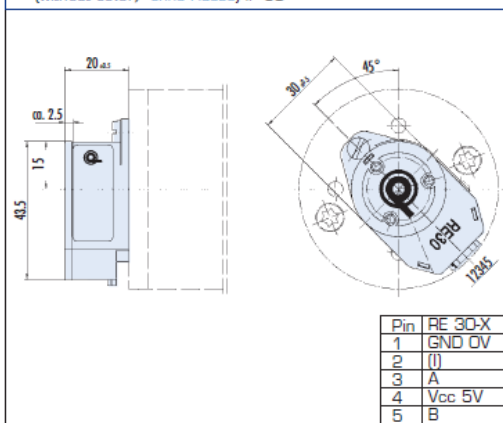
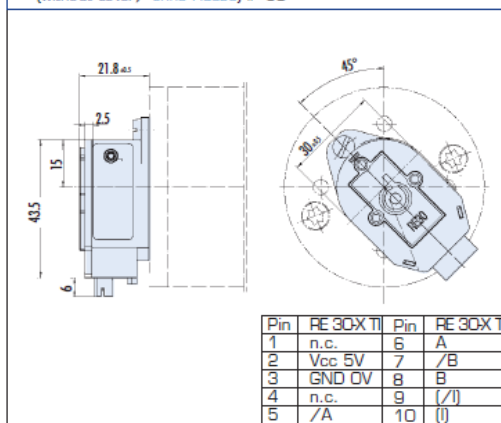
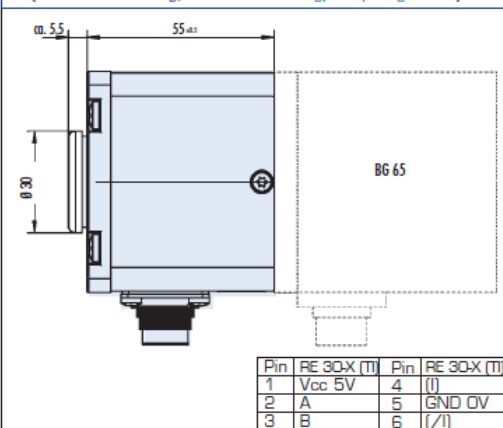
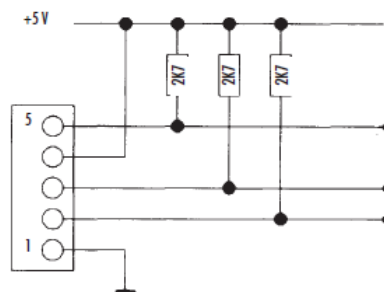
Zum Schutz vor äußeren Einflüssen empfiehlt sich die Verwendung einer IP54-Schutzhaube. In Kombination mit dem BG 65 sind die Inkrementalgeber auch im IP65-Strangpressprofilgehäuse erhältlich.

Data / Leistungsdaten		RE 30-2	RE 30-3	RE 30-3 TI	RE 56-3	RE 56-3 TI
Operating voltage/ Versorgungsspannung	VDC	5	5	5	5	5
Impulses per revolution/ Impulszahl pro Umdrehung	ppr	100 ... 512	500 ... 512	500 ... 512	1000	1000
Signal rise time/ Signalanstiegszeit	ns	200	180	180	180	180
Signal decay time/ Signalabfallzeit	ns*	50	40	40	40	40
Current consumption/ Stromaufnahme	mA	17 (max. 40)	57 (max. 85)	max. 85	57 (max. 85)	max. 85
Output voltage/ Ausgangsspannung (low-level)	VDC	max. 0.4 (3.2 mA)	max. 0.4 (3.9 mA)	max. 0.5 (20 mA)	max. 0.4 (3.9 mA)	max. 0.5 (20 mA)
Output voltage/ Ausgangsspannung (high-level)	VDC	min. 2.4 (40 µA)	min. 2.4 (200 µA)	min. 2.4 (200 µA)	min. 2.4 (200 µA)	min. 2.4 (200 µA)
Max. output current/ max. Ausgangsstrom	mA	-	-	70	-	70
Operating temperature/ Betriebstemperatur	°C	-40 ... +100	-40 ... +100	-40 ... +100	-40 ... +100	-40 ... +100
Protection class/ Schutzart	IP	30	30	30	30	30

*) $C_L = 25\text{pF}$; $R = 11\text{k}\Omega$

RE 30/RE 30 TI; RE 56/RE 56 TI *Wiring suggestions / Schaltungsvorschlag*

Clockwise/counter-cw. detection / Rechts-/Links-Erkennung

Pulse doubling / Impuls-Verdoppelung

Incremental Encoders for **BLDC Motors** Inkrementalgeber für **BG-Motoren**

RE 30
(without cover/ ohne Haube) IP 30

RE 30 TI
(without cover/ ohne Haube) IP 30

RE 30/ RE 30 TI
(with BG 65 housing/ mit BG 65 Strangpressprofilgehäuse) IP 65

RE 30-3
(Connection example/ Beschaltungsvorschlag)


MOTOR

GR 42, 15 - 20 W



Versions of GR 42 / Ausführungen GR 42	P/S
With gearbox / Als Getriebemotor	37
With brake / Als Bremsmotor	48
With controller / Mit Regelelektronik	54
With tachogenerator / Mit Tachogenerator	50
With magnetic pulse generator / Mit magnetischem Impulsgeber	51
With incremental encoder / Mit Inkrementalgeber	52

□ Standard/Standard □ On request/auf Anfrage

- General information about the characteristics of our commutated motors, see page 8
- the standard version has leads (300 mm)
- special windings available on request
- different shaft lengths or shaft on both sides available as per our program
- protection class IP 50, higher class available on request
- ball bearing in the motor shaft. For projects the motor is also available with slide bearing (G 42)



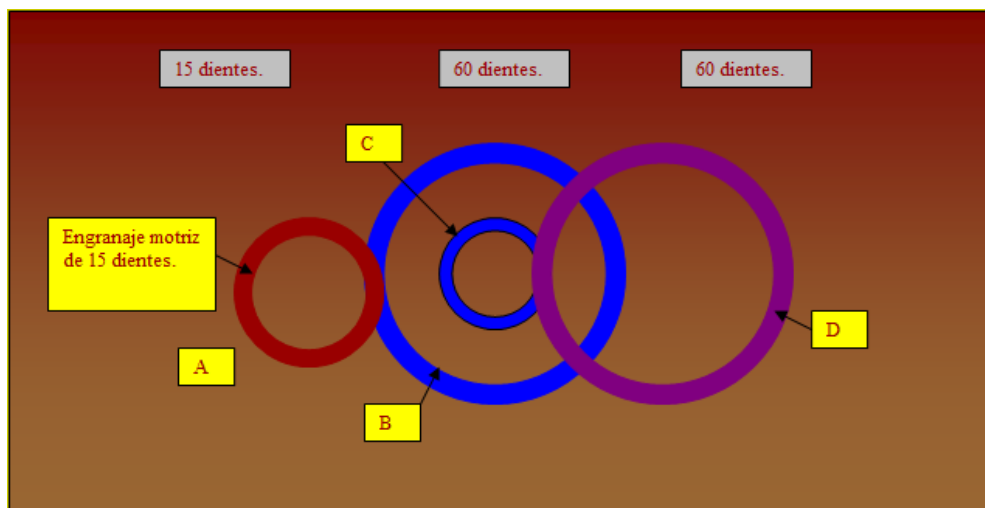
- Allgemeine Informationen über die Eigenschaften unserer Kollektormotoren siehe S. 8
- Der Motor wird standardmäßig mit Litzen (300 mm) geliefert
- Sonderwicklungen auf Anfrage erhältlich
- Auf Anfrage verschiedene Wellenlängen bzw. beidseitige Wellen gemäß unserem Programm lieferbar
- Schutzart IP 50, auf Anfrage auch höher
- Die Motorwelle ist kugellagert. Projektbezogen ist der Motor auch mit Gleitlager erhältlich (G 42)

Data / Leistungsdaten		GR 42x25		
Rated voltage/ Nennspannung		12 VDC	24 VDC	40 VDC
Continuous rated speed/ Nennndrehzahl	rpm*)	3450	3600	3700
Continuous rated torque/ Nennndrehmoment	Ncm*)	3.9	3.8	3.9
Continuous current/ Nennstrom	A *)	1.9	0.9	0.6
Starting torque/ Anlaufmoment	Ncm**)	19	20	22
Starting current/ Anlaufstrom	A**)	7.8	4	2.76
No load speed/ Leerlaufdrehzahl	rpm**)	4350	4200	4400
No load current/ Leerlaufstrom	A**)	0.34	0.17	0.11
Demagnetization current/ Entmagnetisierstrom	A**)	14	6.5	4.1
Rotor inertia/ Trägheitsmoment	gcm ²	71	71	71
Weight of motor/ Motorgewicht	g	390	390	390

Data / Leistungsdaten		GR 42x40		
Rated voltage/ Nennspannung		12 VDC	24 VDC	40 VDC
Continuous rated speed/ Nennndrehzahl	rpm*)	3750	3100	3400
Continuous rated torque/ Nennndrehmoment	Ncm*)	5.3	5.7	5.7
Continuous current/ Nennstrom	A *)	2.7	1.2	0.8
Starting torque/ Anlaufmoment	Ncm**)	32	33	36
Starting current/ Anlaufstrom	A**)	13.2	5.68	3.97
No load speed/ Leerlaufdrehzahl	rpm**)	4550	3800	3950
No load current/ Leerlaufstrom	A**)	0.44	0.18	0.12
Demagnetization current/ Entmagnetisierstrom	A**)	24	10.5	6.3
Rotor inertia/ Trägheitsmoment	gcm ²	110	110	110
Weight of motor/ Motorgewicht	g	490	490	490

*) $\Delta\theta_w = 100\text{ K}$; **) $\theta_R = 20^\circ\text{C}$

REDUCCIÓN DE 16:1



Tren de engranajes compuesto

En el diagrama puede verse el tren de engranajes compuesto. Observa que se usan cuatro engranajes y que los engranajes B y C están sujetos al mismo eje. Cuando el engranaje motriz A da una vuelta completa, el engranaje B girará un cuarto de una vuelta. Ahora bien, como el engranaje C está sujeto al mismo eje que el engranaje B, también da un cuarto de vuelta. Por tanto, el engranaje D solamente girará $1/4$ de $1/4$ de una vuelta, es decir, $1/16$ de una vuelta. Por tanto, la relación de transmisión de este tren de engranajes compuesto es de 16: 1.

Cálculo de la relación de transmisión (tren de engranajes compuesto).

Para calcular la relación de transmisión de un tren de engranajes compuesto, emplea la ecuación siguiente:

Relación de transmisión = $(n.^{\circ} \text{ de dientes de B} / n.^{\circ} \text{ de dientes de A}) * (n.^{\circ} \text{ de dientes de D} / n.^{\circ} \text{ de dientes de C})$.

$$\text{Relación de transmisión} = 60/15 \times 60/60 = 4/1 \times 1/1 = 4/1 = 16/1 \quad 16:1$$

FIBRE-OPTICAL DEVICE

FESTO

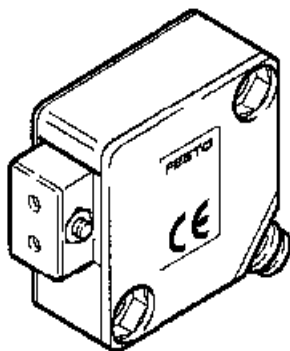

Catalogue page

Part no.: 165327

Page:1

Fibre-optic device SOEG-L-Q30-P-A-S-2L

Optoelectronic sensors



Optoelectronic sensors

Variants

Size

- M12x1 external thread
- M18x1 external thread
- Rectangular design

- Voltage: 10 ... 30 V DC
- Choice of NPN or PNP output
- Plug or cable connection

- Diffuse light sensor, cylindrical or rectangular design
- Retro-reflective sensors, cylindrical or rectangular design
- Reflectors
- Through-beam sensors, cylindrical or rectangular design
- Fibre optic units, rectangular design
- Fibre optic cables

Features

- Ranges to 6000 mm
- IP 65 protection

Accessories:

- Mounting bracket for optical sensors with rectangular design

- Cutting tool SOES-LKS for polymer fibre optic cable

The fibre optic cable is guided within the cutter to ensure a clean, right-angle cutting surface, thus keeping light losses to a minimum. In order to obtain the highest-quality cuts, each hole should be used once only.

Sensor tester SM-TEST-1

The sensor tester is used to test and adjust sensors and proximity switches. The sensor tester facilitates commissioning and service work.

- Voltage supply for testing operation of proximity switches
- Adjustment of proximity switches while attached to cylinders
- Identification of switching outputs of proximity switches and sensors with PNP, NPN, NC and NO functions by means of

FESTO


Catalogue page

Part no.: 165327

Page:2

the appropriate
LED.

Retro-reflective sensors

Sensors are equipped with polarizing filters, assuring that they only respond to light returned by special reflectors. These are based upon the triple mirror principle. The choice of the most suitable reflector for a given application is governed by the required working range and available mounting facilities

.Fibre optic cable

A fibre optic cable can consist of a bundle of glass fibres, or one or more plastic fibres. The function of a fibre optic cable is to guide light from one place to another, even around corners. This is made possible by exploiting the phenomenon of total internal reflection. Total internal reflection occurs whenever light from a material with a high refractive index impinges on the boundary between this material, and a medium with a lower refractive index at an angle less than the maximum angle for total internal reflection. The fibres consist of a core (with a high refractive index) and a sheath (with a low refractive index). Light is constantly reflected back and forth within this construction as a result of total internal reflection, and is thus even able to traverse curved paths.

Installation

Optoelectronic sensors must not be allowed to interfere with each other's operation. A certain minimum distance must be maintained between sensors. This distance depends principally on the sensitivity to which the sensors have been set. For sensors fitted with fibre optic cables, the distance is heavily dependent upon the type of utilised fibre optic cable.

Alignment

Through-beam sensors

- First position the receiver as desired and secure it.
- Then align the transmitter as accurately as possible to the

FESTO



Catalogue page

Part no.: 165327

Page:3

receiver.

Retro-reflective sensors

- First position the reflector as desired and secure it.
- Cover the reflector so that only the centre remains exposed (25% of reflector's surface area).
- Install the retro-reflective sensor such that reliable switching operation is obtained.
- Finally, remove the cover from the reflector.

Diffuse sensors

- Align the sensor to the object to be scanned such that reliable operation is obtained.
- In order to obtain reliable operation, the operating reserve must be active.

Operating reserve

Operating reserve is a measure of the excess radiant energy which falls onto the light-gathering surface, and is evaluated by the light

receiver. Operating reserve may diminish over a period of time due to contamination, changing reflection factor of the object to be scanned and ageing of the transmitter diode, so that reliable operation is no longer assured.

Certain sensors are equipped with a second LED (green) which lights up when approx. 80% of the sensor's available working range is being utilised. With certain other sensors, a yellow LED flashes when available operating reserve is insufficient. This allows for prompt recognition of inadequate operating reliability.

Operating reserve switching hysteresis

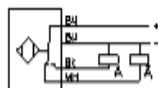
Correction factors

The specified working ranges for diffuse sensors are determined using test cards (Kodak Gray Cards). For other surfaces, the switching point should be determined by applying the listed correction factors.

Working range

The specified working range is the maximum possible distance between the transmitter and receiver (through-beam sensor). To obtain

FESTO



Catalogue page

Part no.: 165327

Page:4

this maximum, the potentiometer must be set to MAX and the specified reflector (retro-reflective sensor) must be used.

Switching functions

Dark switching

A "dark switching" function means that the respective output conducts current (i.e. is activated) when no light strikes the receiver. This is equivalent to a normally closed function (NC).

Light switching

A "light switching" function means that the respective output conducts current (i.e. is activated) when light strikes the receiver. This is equivalent to a normally open function (NO).

Parallel connection

It is possible to connect optoelectronic sensors in parallel to obtain any desired logic functions.

- Current consumption increases
- Inverse currents are cumulative, with the result that impermissibly large voltage drops may occur across the load even if the sensors are non-conductive.



SOEG-L-Q30-P-A-S-2L

Fibre-optic device

Data sheet

Part no.: 165327

Page:1

Feature	Data/description
EU conformity (CE)	CE
Note on EU conformity	Electromagnetic compatibility
Signal processing (measuring principle)	red light
Switch triggering	Reflex/Interrupt
Function on actuation	sender and receiver
Output potential (el. output)	PNP
Coverage range max.	120 mm
Minimum ambient temperature	-5 °C
Maximum ambient temperature	55 °C
Air connection type elec.	Plug
Thread for connector	M 8x1
Number of pins, plug connection	4
Operating status display	Yellow LED
Short-circuit strength	Pulsed
Protection against incorrect polarity	built-in
Type of mounting	Hole
Material of housing	PBT-reinforced
Product weight	0,018 kg
Voltage type	DC
Nominal operating voltage [DC]	24 V
Operating voltage min. (DC)	10 V
Operating voltage max. (DC)	30 V
Idle current max.	25 mA
Maximum switching frequency	1000 Hz
Degree of protection	IP65

Festo KG
Postfach
D-73726 Esslingen
Tel.: (0711) 347-0

9504NH

Lichtleitergerät
Fibre-optic unit
Fibra óptica

Emetteur
Optofiberglävare
Unità a fibre ottiche

CE

FESTO

SOEG-L-Q30-PA-S-2L

165 327



- Nicht für den Einsatz als berührungssichere Schutzvorrichtung! Elektrische Spannung! Vor Arbeiten an der Elektrik: Spannung ausschalten.
- Do not use as contactless protective device! A.C. voltage! Prior to working on electrics: Switch off voltage.
- No utilizar como sensor de protección! ¡Tensión eléctrica! Desconectar la tensión antes de manipular el sistema eléctrico.
- Ne convient pas pour une utilisation en tant que dispositif de sécurité! Tension électrique! Avant toute intervention sur le système électrique: mettre hors tension.
- Får ej användas som beröringsfri skyddsanordning! Elektrisk spänning! Innan arbeten på elektroniken utförs skall spänningen frångöras.
- NON utilizzare come barriera di protezione elettrica. Prima di intervenire sulla parte elettrica, togliere la tensione.

Anwendungshinweise

- Lichtleiter ausrichten
align the cable
LED-
Out-Signal
as Target
mit Objekt im
Lichtstrahl
im Lichtstrahl
als Schranke
ohne Objekt
im Lichtstrahl
LED-
Out-Signal
(Functionstest)
- Maximal eingestellte Tastweite mit Potentiometer reduzieren:
- nur bei fehlender Reaktion aufgrund durchscheinender Objekte oder Ansprechen auf Objekte hinter dem Zielobjekt
- max. 12. Umdrehungen
3. Lichtleitergerät nur reinigen mit:
- Wasser (max. 60 °C) oder Isopropylalkohol

Notes regarding use

- Align fibre-optic cable
Application
Alignement
LED-accepted signal
as sensor
with object in light beam
without object in light beam
as barrier
Yellow (status display) and green (reflective emitter range) signal
- Reduce maximum set sensing distance via potentiometer:
- max. 12 revolutions
- only if no reaction due to transparent objects or responding to object behind target object
3. To clean fibre-optic unit
- use water (max. 60 °C) or isopropyl-alcohol only

Elektrischer Anschluss Electrical connection

Bin Bk Wh	Brown Black White	Monon Azul Negro Blanco
Bin Bk Wh	Brown Black White	Monon Azul Negro Blanco

Indicaciones de utilización

- Alinear la fibra óptica
Utilización
Alineación
LED de salida aceptada
Como sensor
Con objeto en el haz de luz
Sin objeto en el haz de luz
Como barrera
sin objeto en el haz de luz
Ausita (indicación de estado) y verde (nivel de funcionamiento)
- Reducir la distancia máxima de recepción utilizando el potenciómetro:
- únicamente si no responde a los objetos que atraviesan el haz o si responde con objetos situados detrás del objeto enfocado
- máximo 12 vueltas
3. Limpiar la fibra óptica únicamente con:
- agua (máximo 60 °C) o alcohol isopropílico

Instructions d'utilisation

- Aligner l'émetteur
Utilisation
Alignement
LED signal bon
comme capteur
avec objet dans le rayon lumineux
sans objet dans le rayon lumineux
comme barrière
sans objet dans le rayon lumineux
jaune (indication d'état) et vert (niveau de fonctionnement)
- A l'aide du potentiomètre, réduire la distance de détection maximum:
- uniquement en cas d'absence de réaction due à des objets transparents ou de réaction à des objets situés derrière l'objet cible
- max. 12 tours
3. Pour nettoyer l'émetteur, utiliser uniquement:
- de l'eau (max. 60 °C) ou de l'isopropanol

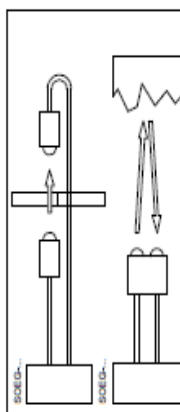
Skötselanvisningar

- Ställ in fiberoptiken
Användning
Inställning
LED-
OK-signal
som sensor
med objekt i ljusstrålen
utan objekt i ljusstrålen
som barriär
med objekt i ljusstrålen
utan objekt i ljusstrålen
Sul (statusindikator) och grön (funktionstest)
- reducera max. inställt avkänningsavstånd med potentiometer:
- endast då reaktion uteblir på grund av genomskinliga objekt eller reaktion från objekt bakom målobjektet
- max. 12 varv
3. Optofiberglävaren (Givaren för fiberoptik) får endast rengöras med:
- vatten (max. 60 °C) eller isopropylalkohol

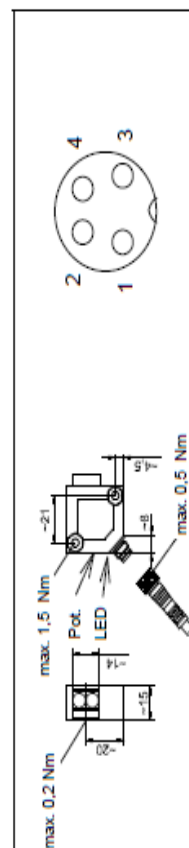
Indicazioni per l'utilizzo

- Allineare l'unità a fibre ottiche
Impiego
Allineamento
Segnale LED di funzione
come sensore a riflessione
con oggetto nel fascio luminoso
senza oggetto nel fascio luminoso
come sensore a trasmissione
senza oggetto nel fascio luminoso
verde (indicazione di stato) e verde (stabilità di funzionamento)
- Ridurre con il potenziometro l'ampiezza di rilevamento max.
- solo in caso di assenza di reazione a causa di oggetti trasparenti oppure di attivazione per oggetti situati dietro l'oggetto da rilevare
- max. 12 rotazioni
3. Pulire l'unità a fibre ottiche solo con:
- acqua (max. 60 °C) oppure alcool isopropilico

El-anslutningar Collegamento elettrico



Mekaniska anslutningar Collegamento meccanico



Conexión mecánica Raccordement mécanique

Mechanischer Anschluss Mechanical connection

Technische Daten	Technical data	Datos técnicos
Betriebsspannungsbereich Max. zul. Restwelligkeit Leerlaufstrom Benennungsbetriebsstrom Spannungsfall Ansprechzeit Abfallzeit Kurzschlussfestigkeit Verpolungsfestigkeit Max. Schaltkapazität Umgebungstemperatur Lagertemperatur Schutzart max. Anzugsrehmoment Werkstoff optisches Fenster Kabelmaterial	10 ... 30 V DC (=Ub) max. ± 20 Ub < 25 mA (incl. LED) 200 mA < 2 V 0,5 ms 0,5 ms ja ja 250 nF -5 °C ... 55 °C -40 °C ... 85 °C IP 65 1,5 Nm PBTP Außenmantel PUR	10 ... 30 V c.c. (=Ub) max. ± 20 Ub < 25 mA (incl. LED) máx. 200 mA < 2 V 0,5 ms 0,5 ms Si Si 250 nF -5°C ... 55°C -40°C ... 85°C IP 65 1,5 Nm PBTP Revestimiento exterior i
Caractéristiques techniques	Tekniska data	Dati tecnici
Plage de tension de service Ondulation résiduelle, adm. Courant à vide Courant de commut. adm. Chute de tension Temps de réponse Temps de chute Protection c. courts-circuits Dérompage Capacité commutable max. Température ambiante Température de stockage Protection Couple de serrage max. Matériau de la fenêtre optique Matériau du câble	10 ... 30 V DC (=Ub) max. ± 20 % Ub < 25 mA (LED comprise) max. 200 mA < 2 V 0,5 ms 0,5 ms oui oui 250 nF -5 °C ... 55 °C -40 °C ... 85 °C IP 65 1,5 Nm PBTP Enveloppe ext. PUR	10 ... 30 V DC (=Ub) max. ± 20 % Ub < 25 mA (incl. LED) 200 mA < 2 V 0,5 ms 0,5 ms si si 250 nF -5 °C ... 55 °C -40 °C ... 85 °C IP 65 1,5 Nm PBTP guaina esterna poliuretano

FIBRE-OPTIC CABLE

FESTO

Catalogue page

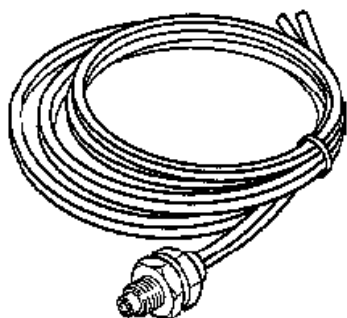
Part no.: 165358

Page:1

Fibre-optic cable

SOEZ-LLK-RT-2,0-M6

Optoelectronic sensors



Optoelectronic sensors

Variants

Size

- M12x1 external thread
- M18x1 external thread
- Rectangular design

- Voltage: 10 ... 30 V DC
- Choice of NPN or PNP output
- Plug or cable connection

- Diffuse light sensor, cylindrical or rectangular design

- Retro-reflective sensors, cylindrical or rectangular design
- Reflectors

- Through-beam sensors, cylindrical or rectangular design

- Fibre optic units, rectangular design
- Fibre optic cables

Features

- Ranges to 6000 mm
- IP 65 protection

Accessories:

- Mounting bracket for optical sensors with rectangular design

- Cutting tool SOES-LKS for polymer fibre optic cable

The fibre optic cable is guided within the cutter to ensure a clean, right-angle cutting surface, thus keeping light losses to a minimum. In order to obtain the highest-quality cuts, each hole should be used once only.

Sensor tester SM-TEST-1

The sensor tester is used to test and adjust sensors and proximity switches. The sensor tester facilitates commissioning and service work.

- Voltage supply for testing operation of proximity switches
- Adjustment of proximity switches while attached to cylinders
- Identification of switching outputs of proximity switches and sensors with PNP, NPN, NC and NO functions by means of

FESTO

Catalogue page

Part no.: 165358

Page:2

the appropriate
LED.

Retro-reflective sensors

Sensors are equipped with polarizing filters, assuring that they only respond to light returned by special reflectors. These are based upon the triple mirror principle. The choice of the most suitable reflector for a given application is governed by the required working range and available mounting facilities

.Fibre optic cable

A fibre optic cable can consist of a bundle of glass fibres, or one or more plastic fibres. The function of a fibre optic cable is to guide light from one place to another, even around corners. This is made possible by exploiting the phenomenon of total internal reflection. Total internal reflection occurs whenever light from a material with a high refractive index impinges on the boundary between this material, and a medium with a lower refractive index at an angle less than the maximum angle for total internal reflection. The fibres consist of a core (with a high refractive index) and a sheath (with a low refractive index). Light is constantly reflected back and forth within this construction as a result of total internal reflection, and is thus even able to traverse curved paths.

Installation

Optoelectronic sensors must not be allowed to interfere with each other's operation. A certain minimum distance must be maintained between sensors. This distance depends principally on the sensitivity to which the sensors have been set. For sensors fitted with fibre optic cables, the distance is heavily dependent upon the type of utilised fibre optic cable.

Alignment

Through-beam sensors

- First position the receiver as desired and secure it.
- Then align the transmitter as accurately as possible to the



Catalogue page

Part no.: 165358

Page:3

receiver.

Retro-reflective sensors

- First position the reflector as desired and secure it.
- Cover the reflector so that only the centre remains exposed (25% of reflector's surface area).
- Install the retro-reflective sensor such that reliable switching operation is obtained.
- Finally, remove the cover from the reflector.

Diffuse sensors

- Align the sensor to the object to be scanned such that reliable operation is obtained.
- In order to obtain reliable operation, the operating reserve must be active.

Operating reserve

Operating reserve is a measure of the excess radiant energy which falls onto the light-gathering surface, and is evaluated by the light

receiver. Operating reserve may diminish over a period of time due to contamination, changing reflection factor of the object to be scanned and ageing of the transmitter diode, so that reliable operation is no longer assured.

Certain sensors are equipped with a second LED (green) which lights up when approx. 80% of the sensor's available working range is

being utilised. With certain other sensors, a yellow LED flashes when available operating reserve is insufficient. This allows for prompt recognition of inadequate operating reliability.

Operating reserve switching hysteresis

Correction factors

The specified working ranges for diffuse sensors are determined using test cards (Kodak Gray Cards). For other surfaces, the switching point should be determined by applying the listed correction factors.

Working range

The specified working range is the maximum possible distance between the transmitter and receiver (through-beam sensor). To obtain



Catalogue page

Part no.: 165358

Page:3

receiver.

Retro-reflective sensors

- First position the reflector as desired and secure it.
- Cover the reflector so that only the centre remains exposed (25% of reflector's surface area).
- Install the retro-reflective sensor such that reliable switching operation is obtained.
- Finally, remove the cover from the reflector.

Diffuse sensors

- Align the sensor to the object to be scanned such that reliable operation is obtained.
- In order to obtain reliable operation, the operating reserve must be active.

Operating reserve

Operating reserve is a measure of the excess radiant energy which falls onto the light-gathering surface, and is evaluated by the light

receiver. Operating reserve may diminish over a period of time due to contamination, changing reflection factor of the object to be scanned and ageing of the transmitter diode, so that reliable operation is no longer assured.

Certain sensors are equipped with a second LED (green) which lights up when approx. 80% of the sensor's available working range is

being utilised. With certain other sensors, a yellow LED flashes when available operating reserve is insufficient. This allows for prompt recognition of inadequate operating reliability.

Operating reserve switching hysteresis

Correction factors

The specified working ranges for diffuse sensors are determined using test cards (Kodak Gray Cards). For other surfaces, the switching point should be determined by applying the listed correction factors.

Working range

The specified working range is the maximum possible distance between the transmitter and receiver (through-beam sensor). To obtain



Catalogue page

Part no.: 165358

Page:4

this maximum, the potentiometer must be set to MAX and the specified reflector (retro-reflective sensor) must be used.

Switching functions

Dark switching

A "dark switching" function means that the respective output conducts current (i.e. is activated) when no light strikes the receiver. This is equivalent to a normally closed function (NC).

Light switching

A "light switching" function means that the respective output conducts current (i.e. is activated) when light strikes the receiver. This is equivalent to a normally open function (NO).

Parallel connection

It is possible to connect optoelectronic sensors in parallel to obtain any desired logic functions.

- Current consumption increases
- Inverse currents are cumulative, with the result that impermissibly large voltage drops may occur across the load even if the sensors are non-conductive.

**SOEZ-LLK-RT-2,0-M6**

Fibre-optic cable

Data sheet

Part no.: 165358

Page:1

Feature	Data/description
Signal processing (measuring principle)	red light
Switch triggering	Reflex
Function on actuation	Polymer fibre optic cable
Coverage range max.	120 mm
Minimum ambient temperature	-40 °C
Maximum ambient temperature	70 °C
Mounting thread	M 6
Material of housing	brass
Product weight	0,02 kg
Coating of housing	Nickel-plated
Degree of protection	IP65

Festo KG
Postfach
D-73726 Esslingen
Tel.: (0711) 347-0

5904NH

Lichtleiter (Kunststoff)
Fibre-optic cable (polymer)
Fibra óptica (de polímero)

Fibre optique (polymère)
Optokabel (polymer)
Condotore a fibre ottiche
(plastica)



FESTO

SOEZ-LLK-RT-2,0-M6

165 358



- Nicht für den Einsatz als berührungslose Schutzvorrichtung! Elektrische Spannung! Vor Arbeiten an der Elektrik: Spannung ausschalten.
- Do not use as contactless protective device! A.C. voltage! Prior to working on electric: Switch off voltage.
- ¡No utilizar como sensor de protección! ¡Tensión eléctrica! Desconectar la tensión antes de manipular el sistema eléctrico.
- Ne convient pas pour une utilisation en tant que dispositif de sécurité! Tension électrique! Avant toute intervention sur le système électrique: mettre hors tension.
- Får ej användas som beröringsfri skyddsanordning! Elektrisk spänning! Innan arbeten på elektroniken utförs skall spänningen frångöras.
- NON utilizzare come barriera di protezione elettrica. togliere la tensione.

Anwendungshinweise

1. benötigte Lichtleiterlänge bestimmen
2. Lichtleiter nur mit Lichtleiterschneidern kürzen
- Typ SOE-LSK, TNr.36 479
- Jedes Schneidloch nur zum Einmalgebrauch (sonst unsauberer Schnitt => Lichtverluste)
3. Lichtleiter bis zum Anschlag in Steckanschlüsse stecken
4. Klemmschraube festdrehen
5. Steckverbindung von Zug- oder Torsionsbelastung freihalten

Indicaciones de utilización

1. Determinar la longitud necesaria del conductor
2. Cortar únicamente con la herramienta especial
- tipo SOE-LSK, n° de artículo 36 479
- Usar cada segmento cortado una sola vez (en caso contrario puede haber pérdidas de luz)
3. Para encharar el conductor, introducirlo hasta el tope en el conector
4. Ajustar el tornillo de fijación
5. Evitar fuerzas de tracción o torsión en la unión

Skötselanvisningar

1. Nödvändig längd på optokabeln bestäms
2. Kapa optokabeln endast med ett härfor avsett verktyg
- Typ SOE-LSK, art.nr. 36 479
- kapningshål får endast användas en gång (annars ojämnt snitt => ljusförlust)
3. Tryck in optokabeln till anslag i stötkontakt
4. Drag åt låsskruven
5. Håll skankoppling fri från drag- eller vridkrafter

Notes regarding use

1. Determine required length of fibre-optic cable
2. Only use fibre optic cable cutters to shorten fibre-optic cables
- Type SOE-LSK, Pl. No. 36 479
- Each cutting hole for single use only (otherwise cut is inaccurate => loss of light)
3. Insert fibre-optic cable into plug up to stop
4. Securely tighten locking screw
5. Keep plug connection free of tension and torsion

Instructions d'utilisation

1. Déterminer la longueur de fibre optique nécessaire
2. Ne couper la fibre optique qu'avec un cutter pour fibres optiques
- type SOE-LSK, n° de pièce 36 479
- n'utiliser les trous qu'une seule fois (la coupe ne serait sinon pas nette => déperditions de lumière)
3. Engager les fibres optiques jusqu'en butée dans les douilles
4. Bloquer la vis de serrage
5. Éviter toute traction ou torsion du connecteur

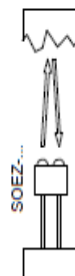
Indicazioni per l'utilizzo

1. Stabilire la lunghezza necessaria del conduttore
2. Tagliare il conduttore alla lunghezza desiderata con l'apposito tronchesino
- Tipo SOE-LSK, Cod. prod. 36479
- Ogni foro del tronchesino è utilizzabile una sola volta (diversamente il taglio non risulta netto, con conseguente dispersione di luce)
3. Inserire il conduttore a fibre ottiche nel connettore fino all'arresto
4. Stringere la vite di serraggio
5. Evitare sollecitazioni di torsione o trazione sul connettore

Elektrischer Anschluß Electrical connection

Conexión eléctrica Raccordement électrique

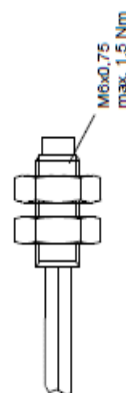
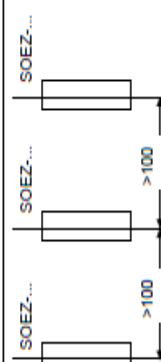
El-anslutningar Collegamento elettrico



Mechanischer Anschluß Mechanical connection

Conexión mecánica Raccordement mécanique

Mekaniska anslutningar Collegamento meccanico



Technische Daten

Erfassungsbereich max.
0 ... 120 mm
Erfassungsbereich min.
0 ... 24 mm
Umgebungstemperatur
-40 °C ... 70 °C
Lagertemperatur
-40 °C ... 70 °C
Schutzart
IP 65
max. Anzugsdrehmoment
1,5 Nm
Werkstoff optische Faser
PMMA
Außenmantel
Polyethylen

gemessen mit Normtestplatte weiß und SOEG-L-...

Technical data

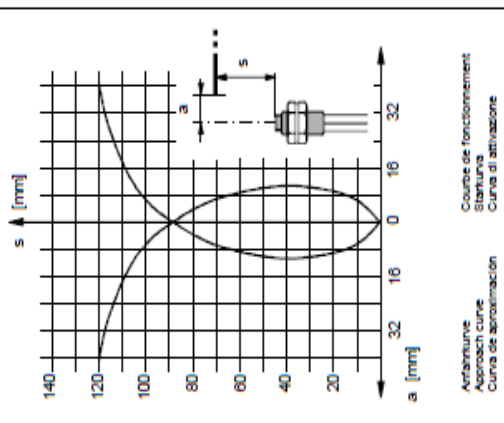
Max. detection range
0 ... 120 mm
Min. detection range
0 ... 24 mm
Ambient temperature
-40 °C ... 70 °C
Storage temperature
-40 °C ... 70 °C
Degree of Protection
IP 65
Max. tightening torque
1,5 Nm
Material of fibre optic
PMMA
Outer casing
Polyethylene

measured with standard test plate - white and SOEG-L-...

Datos técnicos

Distancia máx. de detección
0 ... 120 mm
Distancia mín. de detección
0 ... 24 mm
Temperatura - ambiente
-40 °C ... 70 °C
- de almacenamiento
-40 °C ... 70 °C
Tipo de protección
IP 65
Par de apriete máximo
1,5 Nm
Material de la fibra óptica
PMMA
Revestimiento exterior
Poliétileno

Medición efectuada con placa normalizada blanca y SOEG-L-...



Caractéristiques techniques

Plage de détection max.
0 ... 120 mm
Plage de détection min.
0 ... 24 mm
Température ambiante
-40 °C ... 70 °C
Température de stockage
-40 °C ... 70 °C
Protection
IP 65
Couple de serrage max.
1,5 Nm
Matériau de fibre optique
PMMA
Matériau du câble
Polyéthylène

mesuré avec une plaque de mesure normalisée blanche et SOEG-L-...

Tekniska data

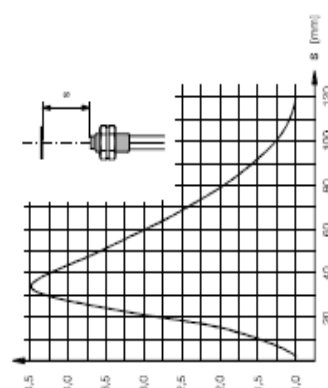
Max. registreringsområde
0 ... 120 mm
Min. registreringsområde
0 ... 24 mm
Omgivningstemperatur
-40 °C ... 70 °C
Lagringstemperatur
-40 °C ... 70 °C
Skyddsklass
IP 65
Max. ådragningsmoment
1,5 Nm
Material optokabel
PMMA
Ytterskikt
Polyetylen

uppmätt med vit normplatta och SOEG-L-...

Dati tecnici

Intervallo rilevamento max.
0 ... 120 mm
Intervallo rilevamento min.
0 ... 24 mm
Temperatura ambiente
-40 °C ... 70 °C
Temperatura di stoccaggio
-40 °C ... 70 °C
Grado di protezione
IP 65
Coppia max. di serraggio
1,5 Nm
Materiale fibre ottica
PMMA
Guaina Esterna
Poliethylene

misurato con piastra di misura a norme, bianca e SOEG-L-...



Proximity sensors SIED-..., inductive

FESTO

Technical data

Ordering data – M12x1				
Installation		Electrical connection		Part No. Type
Flush	Non-flush	Cable	Plug	
NO contact				
■	–	■	–	538 272 SIED-M12B-ZS-K-L
■	–	–	■	538 271 SIED-M12B-ZS-S-L
–	■	■	–	538 268 SIED-M12NB-ZS-K-L
–	■	–	■	538 267 SIED-M12NB-ZS-S-L
NC contact				
■	–	■	–	538 274 SIED-M12B-ZO-K-L
■	–	–	■	538 273 SIED-M12B-ZO-S-L
–	■	■	–	538 270 SIED-M12NB-ZO-K-L
–	■	–	■	538 269 SIED-M12NB-ZO-S-L

General technical data						
Size			M12x1	M18x1	M30x1.5	
Type of installation			flush or non-flush			
Nominal switching distance S _n	flush	[mm]	2.0	5.0	10.0	
	non-flush	[mm]	4.0	8.0	15.0	
Assured switching distance S _a	flush	[mm]	1.62	4.05	8.1	
	non-flush	[mm]	3.24	6.48	12.15	
Repetition accuracy	flush	[mm]	0.04	0.1	0.2	
	non-flush	[mm]	0.08	0.16	0.3	
Type of mounting			Via lock nut			
Tightening torque			[Nm]	1.0	2.0	5.0
Ready status display			–			
Switching status display			Yellow LED			
Conforms to			DIN EN 60947-5-2			

Electrical data					
Size		M12x1	M18x1	M30x1.5	
Switch output		PNP or NPN			
Switching element function		NO contact			
Electrical connection	Cable	3-core			
Cable length	[m]	2.5			
Operating voltage range	[V DC]	10 ... 30			
Residual ripple	[%]	10			
Max. switching frequency	flush	[Hz]	2000	1000	500
	non-flush	[Hz]	2000	1000	500
Max. output current		[mA]	200		
Voltage drop		[V]	≤ 1.8		
Idle current		[mA]	≤ 15		
Protection against short circuit		Pulsed			
Protection against polarity reversal		For all electrical connections			
Resistance to interference from magnetic fields		–			
Protection class to EN 60 529		IP65/IP67			
CE symbol		89/336/EEC (EMC)			

Proximity sensors SIEN-...-PA, inductive

FESTO

Technical data

Reduction factors of nominal switching distance S_n			
Size	M12x1	M18x1	M30x1.5
Steel St 37	1.0		
Stainless steel St 18/8	0.6 ... 1.0		
Brass	0.35 ... 0.5		
Aluminium	0.35 ... 0.5		
Copper	0.25 ... 0.45		

Materials			
Size	M12x1	M18x1	M30x1.5
Housing	Polyamide, reinforced		
Cable sheath	Polyvinyl chloride		
Note on materials	Free of copper, PTFE and silicone		

Operating and environmental conditions			
Size	M12x1	M18x1	M30x1.5
Ambient temperature [°C]	-25 ... +70		
Ambient temperature with flexible cable installation [°C]	0 ... +70		
Corrosion resistance class CRC ¹⁾	4		

Weight [g]			
Size	M12x1	M18x1	M30x1.5
	113	127	158

I/O CONNECT

- Power supply 24V: $V_{nom} = 24V$ (30V if automatic charger is connected)
- Analog inputs: $V_{in}=0-10V$ / $I_{min} > 1mA$
- Digital inputs : $V_{in}=0/24V$ / $I_{min} > 1mA$ (trigger level ON $U_{in} > 8,5V$ / trigger level OFF $U_{in} < 5,7V$)
- Digital outputs : $U_{Anom.} = 24V$ / $I_{Amax} = 0,3 A$ (short-circuit proof)
- Relais data: 24-30V max. 500mA (NOT short-circuit proof)

WEBCAM



Labtec® Webcam Pro

Artikel Nr.: 961358-0914



Verkaufsargumente

- USB-Webkamera für den PC mit CMOS-VGA-Sensor für bis zu 1280 x 960 Pixel Auflösung bei Standbildern (Software interpoliert)
- Erstellen von Videos und Standbildern
- Versenden von Videos und Bildern in E-Mails ganz einfach per Tastendruck
- Ergänzen von Instant-Messages mit live-Videos. Funktioniert mit gängigen Instant Messaging Programmen, wie Yahoo® Messenger und MSN® Messenger
- Auch als Überwachungskamera einsetzbar
- Integriertes Mikrophon, Auslösertaste, LED-Signal bei aktivierter Kamera, manueller Fokus
- Einfache Installation dank USB-Anschluss
- Zwei Jahre Garantie

Funktionen und Nutzen

Funktionen	Nutzen
Professionelle Auflösung	<ul style="list-style-type: none"> - Videos mit bis zu 640 x 480 Pixel Auflösung - Standbilder mit bis zu 1280 x 960 Pixel Auflösung (Software interpoliert)
Video Instant Messaging	<ul style="list-style-type: none"> - Live-Video-Chat über Instant Messenger Programmen wie MSN®- oder Yahoo®-Messenger
Inklusive Bewegungsmelder-Software	<ul style="list-style-type: none"> - Überwacht Ihr Zuhause, wenn Sie nicht da sind

Paketeigenschaften

	<i>Stück</i>	<i>Karton</i>
<i>Art. #</i>	961358-0914	
<i>UPC/ EAN</i>	5099206 968202	50992069682059
<i>Gewicht</i>	547 g	4,00 kg
<i>Breite</i>	19,60 cm	36,40 cm
<i>Tiefe</i>	7,80 cm	22,40 cm
<i>Höhe</i>	25,00 cm	40,70 cm
<i>Volumen</i>	3,8 dm ³	0,03319 m ³
<i>Anzahl im Karton</i>	10	
<i>Anzahl Palette</i>	520	52
<i>Garantie</i>	Zwei Jahre	
<i>Herkunftsland</i>	China	

Paketinhalt

- Hardware:
 - Labtec® Webcam Pro
 - USB-Verbindungskabel
 - Kamera-Standfuß für stabilen Halt
- Software auf CD:
 - Videos mit bis zu 640 x 480 Pixel Auflösung, 30 Frames Pro Sekunde
 - Standbilder mit bis zu 1280 x 960 Pixel Auflösung (Software interpoliert)
 - Bewegungsmelder
 - Video- und Bild-Versand per e-mail – auf einen Klick
 - Fotoalbum
 - Microsoft® NetMeeting

Systemvoraussetzungen

- PC mit 500 MHz Pentium III oder höher
- Windows® 98, Me, 2000 oder XP
- 128 MB Arbeitsspeicher
- 75 MB freier Festplattenspeicher
- Windows kompatible Soundkarte (empfohlen: Full Duplex)
- Lautsprecher für Audio
- Freier USB-Anschluss
- CD-ROM Laufwerk
- Zum Versenden von Video-e-Mails: e-Mail Programm, das Attachments unterstützt (z.B. AOL 4.0, Microsoft Outlook Express oder Netscape mail). 14,4 Kbps Modem, Internet-Verbindung
- Für Internet-Videokonferenz: 28.8 Kbps Modem oder mehr

ANEXO 4

Materiales utilizados para la construcción del entono

Lista de materiales utilizados

En la tabla se muestran todos los materiales utilizados en la construcción del entorno de trabajo para que el Robotino realice las tareas de búsqueda del camino más óptimo en tiempo real.

Tipo de material	Empleados para:
Cinta de aluminio	Caminos (arcos)
Cartulina de color negro	Nodos
Plancha de madera	Ubicar arcos y nodos
Códigos de barra impresos (code 128)	Identificador del camino
Lámparas de luz blanca	Iluminar el entorno de trabajo